LOAN DOCUMENT

PHOTOGRAPH THIS SHEET

DTIC ACCESSION NUMBER

LEVEL

INVENTORY

DOCUMENT IDENTIFICATION

DISTRIBUTION STATEMENT

HANDLE WITH CARE

ACCESSION FOR
NTIS GRA&I
DTIC TRAC
UNANNOUNCED
JUSTIFICATION

BY

DISTRIBUTION/

AVAILABILITY CODES

DISTRIBUTION	AVAILABILITY AND/OR SPECIAL	
A-1		

DISTRIBUTION STAMP

DATE ACCESSIONED

DATE RETURNED

19981223 070

DATE RECEIVED IN DTIC

REGISTERED OR CERTIFIED NUMBER

PHOTOGRAPH THIS SHEET AND RETURN TO DTIC-FDAC

DTIC FORM 70A JUN 90

DOCUMENT PROCESSING SHEET

PREVIOUS EDITIONS MAY BE USED UNTIL STOCK IS EXHAUSTED.

LOAN DOCUMENT

APPENDIX 12

AUXILIARY BUS INTERFACE (ABI) MANAGEMENT

FINAL SOFTWARE REPORT

DATA ITEM NO. A005

INTEGRATED ELECTRONIC WARFARE SYSTEM
ADVANCED DEVELOPMENT MODEL (ADM)

PREPARED FOR:

NAVAL AIR DEVELOPMENT CENTER
WARMINSTER, PENNSYLVANIA

CONTRACT N62269-78-C-0078

RAYTHEON
ELECTROMAGNETIC
SYSTEMS DIVISION

1 OCTOBER 1977

APPENDIX 12

AUXILIARY BUS INTERFACE MANAGEMENT DESIGN SPECIFICATION FINAL SOFTWARE REPORT

DATA ITEM A005

INTEGRATED ELECTRONIC WARFARE SYSTEM (IEWS)
ADVANCED DEVELOPMENT MODEL (ADM)

Contract No. N62269-75-C-0070

Prepared for:

Naval Air Development Center
Warminister, Pennsylvania

Prepared by:

RAYTHEON COMPANY
Electromagnetic Systems Division
6380 Hollister Avenue
Goleta, California 93017

1 OCTOBER 1977

RAYTHEON	RAYTHEON COMPANY LEXINGTON, MASS. 02173	CODE IDENT NO. 49956	SPEC NO. 53959-GT-0754
			SHEET 1 OF 97 REV

TYPE OF SPEC

COMPUTER SUBPROGRAM DESIGN DOCUMENT

TITLE OF SPEC

ABI MANAGEMENT, IEWS, ADM

FUNCTION	APPROVED	DATE	FUNCTION	APPROVED	DATE
WRITER	Thomas L. Chernabb	25 Feb 77			

REVISIONS

CHK	DESCRIPTION	REV	CHK	DESCRIPTION	REV

REVISION																										
SHEET NO.																										
REV STATUS OF SHEETS	REVISION																									
	SHEET NO.																									

TABLE OF CONTENTS

1.0 SCOPE

1.1 IDENTIFICATION

This document describes the implementation of the ABI Management 1 and ABI Management 2 Functional Groups of the SC Operational Software resident in the Classification and Analysis Processors, respectively.

1.2 SUBPROGRAM TASKS

1.2.1 ABI Management 1 Driver (AB1DR)

AB1DR shall process two types of executive messages:

1) Analysis requests from the Classification processor, and
2) Aux Bus control messages from the Analysis processor and from the Resource Management processor.

AB1DR shall output one of the following, as required:

1) Null analysis return message destined for the CP.
2) Analysis start message destined for the AP.
3) SPDW control (start or stop) message to the Sorter.

AB1DR and the supporting subroutines shall constitute the ABI Management 1 Functional Group.

1.2.2 ABI Management 2 Driver (AB2DR)

AB2DR shall process Analysis Start messages received from AB1DR. AB2DR shall assign a priority to the message, place the message on one of the analysis queues, and then output a Start AB1DR message.

1.2.3 ABI Initialization Driver (ABIDR)

ABIDR shall search the analysis queues for an analysis start message (starting with the highest priority queue). If found and if sufficient resources are available, the analysis shall be initiated (i.e., 2 Buffers assigned, the Analysis Management Table (AMT) entry and the Analysis Buffer Assignment Table (AAT) entry initialized).

1.2.4 ABI Return Processing (ABFUL and ABRDR)

ABFUL shall be executed each time a Buffer Full interrupt is received by the AP EXEC. It shall enable the co-buffer assigned to the analysis and shall output an Exec message to start ABRDR, the ABI Return driver. ABRDR shall move the data to AP private storage. If this additional data is sufficient to allow completion of the analysis, the buffers shall be disabled and a start ABDDR message shall be output.

1.2.5 ABI Time Out Check (ABTCK)

ABTCK shall be called by the AP EXEC every 50 milliseconds. ABTCK shall enable the buffers for any timed analyses waiting to be enabled. If it is determined that an analysis has timed out, the buffers shall be paused and a Start ABDDR message shall be output.

1.2.6 ABI Done Driver (ABDDR)

ABDDR shall be called whenever an analysis has timed out or the required number of buffers have been processed. If the analysis has timed out, any partial buffers shall be unloaded and the accumulated data processed. The results of the analysis shall then be output via an Analysis Return message for the CP.

2.0 APPLICABLE DOCUMENTS

The following documents, of the exact issue shown, form a part of this specification to the extent specified herein. In the event of conflict between the documents referenced herein and the contents of this specification, the contents of the Computer Program Design Specification for the Integrated Electronic Warfare System (IEWS) Advanced Development Model (ADM) Program shall be considered superseding requirements.

2.1 COMPUTER PROGRAM PERFORMANCE SPECIFICATION

Computer Program Performance Specification for the Integrated Electronic Warfare System (IEWS) Advanced Development Model (ADM) Program (U), Raytheon Company, Electromagnetic Systems Division, (Number 061290529), (date 1 June 1976), (classification U).

2.1.1 Applicable CPPS Paragraphs

AB1	Driver	3.3.5.1
AB2	Driver	3.3.5.1
ABI	Initialization Driver	3.3.5.1
ABI	Return Driver	3.3.5.1
ABI	Time Out Check	3.3.5.1
ABI	Done Driver	3.3.5.1 thru 3.3.5.6

2.2 COMPUTER PROGRAM DESIGN SPECIFICATION

Computer Program Design Specification for the Integrated Electronic Warfare System (IEWS) Advanced Development Model (ADM) Program (U), Raytheon Company, Electromagnetic Systems Division, (Number 53959-GT-0750), (date TBD), (classification U).

2.3 DATA BASE DESIGN DOCUMENT

The Common Data Base Design Document, System Controller Unit, IEWS, ADM, document No. 53959-GT-0751, shall apply to this subprogram.

2.4 MISCELLANEOUS DOCUMENTS

The following documents shall apply to this subprogram:

Document No.	Document Title
53959-GT-0756	Computer Subprogram Design Document, Executive, IEWS, ADM
53959-JK-1002	Interface Control Document, System Controller-Sorter
53959-GT-0759	Computer Subprogram Design Document, Data Extraction, IEWS, ADM
WS-8506 Revision 1, 1 November 1971	Requirements for Digital Computer Program Documentation
5413:IEWS:76:33	IEWS System Controller Aux Bus Interface Design Specification

3.0 REQUIREMENTS

3.1 SUBPROGRAM DETAILED DESCRIPTION

3.1.1 ABI Management 1 Driver (AB1DR)

AB1DR shall be called to process Analysis Request messages from the CP and Aux Bus Control messages from both the AP and the RMP. AB1DR shall receive from the CP EXEC a pointer to such a message in the X-register. If the message is not an analysis request, processing shall continue at label AB1D5Ø. Otherwise, the AW flag of the Analysis Request shall be tested. AW equal to zero shall be interpreted as a null analysis request. A null analysis return message shall be output via EXMSG and control returned to the CP EXEC.

If the AW flag is set, the analysis request is a request for actual analysis data and as such shall be relayed to the AP in the form of an Analysis Start message. Also, the value of peak amplitude, EFPAMP, for the emitter under analysis shall be transferred from the Emitter File (EF) to the AB1 Management Table (AUXMT). ABIDR shall use peak amplitude to generate an Aux Bus Control (SPDW start) message. Control shall then be transferred to the CP EXEC.

3.1.1.1 AB1D5Ø

Aux Bus Control messages from both the AP and RMP shall be processed at label AB1D5Ø. "Stop SPDW'S" Aux Bus Control messages shall be transformed into Sorter Control messages and output via EXMSG to the Sorter, unless the SPDW's are still required by the other processor. RMP and AP SPDW activity shall be recorded in the AUXMT. "SPDW Request" Aux Bus Control shall always result in a Sorter Control message being output. However, the RMP shall have priority over the data flow on the Aux Bus. Therefore, if the request is

from the AP and the RMP has already requested SPDW's, the message output to Sorter shall have the TTAMP required by the RMP. If the request is from the RMP and the AP has already requested SPDW's, the TTAMP from the RMP's request shall be used. In either case the correct combination of the SC and AGTC flags (AP and RMP, respectively) shall be set in order to keep SPDW's flowing to the AP and Technique Generator as required.

3.1.2 ABI Management 2 Driver (AB2DR)

AB2DR shall be called to process Analysis Start messages sent from the CP. AB2DR shall receive from the AP EXEC a pointer to such a message in the X-register. Subroutine AB2AP shall immediately be called to calculate the priority of the start message. The following algorithm shall be used:

1) Priority = 0 If Return Module Code (RMC) is not an update module and the analysis requested is not contemporaneous.

2) Priority = 1 if RMC is not an update module and the analysis requested is contemporaneous.

3) Priority = 2 if RMC is an update module and the analysis requested is not contemporaneous.

4) Priority = 3 if RMC an update module and the analysis requested is contemporaneous.

The priority shall be saved in the analysis queue entry. Subroutine AB2BR shall be called to calculate the number of double-buffers in the 1K RAM required for this analysis. The following algorithm shall be used:

1) If not contemporaneous, double-buffer requirement = 1.

2) If contemporaneous, double-buffer requirement = 1 + C1 + C2 + C3, where Ci = 0 or 1 and Ci = 1 indicates a suspected contemporaneous EFN.

The double buffer requirement shall be saved in the analysis queue entry. Next, an analysis module code (AMC) shall be calculated by examining the flags of the analysis start message:

1) If SA is set, AMC = 0
2) If FA is set, AMC = 1
3) If PA is set, AMC = 2
4) If CA is set, AMC = 3
5) If DI is set, AMC = 4

The AMC shall then be saved in the analysis queue entry, which shall be placed on one of the four analysis queues using the assigned priority as an index. Then, a message shall be sent via EXMSG to start ABIDR, which shall search the analysis queues for entries. Control is then returned to the AP EXEC.

3.1.3 ABI Initialization Driver (ABIDR)

ABIDR shall receive a pointer to a Start-ABIDR MSG (in the X-Register) from the AP EXEC. The pointer shall be saved in a return-block message buffer. The return block msg shall be output via EXMSG, to return the input msg block to the pool of available blocks. ABIDR shall then search the Analysis Queues, starting with the highest priority queue for a re-formatted analysis start message (analysis queue entry format). If found, the ABI Resource Check Subroutine (ABRCK) shall be called to verify the availability of three ABI Management resources:

1) Analysis Mangement Table (AMT) storage.
2) 1K RAM buffer storage
3) Analysis availability of the EFN's to be scrutinized

If sufficient resources are available, the entry (a re-formatted analysis start message) shall be removed from the queue and the analysis initiated via subroutine ABINT. Control shall then be sent to the beginning of ABIDR to re-search queues for another potential initiate. If resources are not sufficient and the priority is high (0 or 1), one of the analysis purge tests (ABPG1 and ABPG2) shall be called to see if an analysis can be purged and the necessary resources freed. If the required purging is not possible, control is returned to the AP EXEC and the high priority (0 or 1) analysis start shall remain queued. If the purging is successful, the analysis queue entry shall be removed from the queue and the analysis initiated via subroutine ABINT. Control shall then be sent to the beginning of ABIDR.

If the resources are not sufficient and the priority is low (2 or 3) the analysis queue entry shall be removed from the analysis queues, and a null analysis return message formatted and output via ABRTN. The searching of the queues shall then be resumed at the next lower level. If all queues are empty, control shall be returned to the AP EXEC.

3.1.3.1 ABI Resource Check (ABRCK) -

ABRCK shall be called to determine the availability of sufficient ABI Management resources to perform the requested analysis. ABRCK shall be called with a pointer to an analysis queue entry in the X-register. Three returns to the calling routine shall be possible:

1) There is insufficient buffer space in the 1K RAM or the AMT is full.

2) An emitter (EFN) necessary for this analysis is already under analysis. The conflicting EFN is returned in the right byte of the A-register.

3) Normal return. Sufficient resources are available.

Return 1 shall be performed if:

1) After incrementing, the total number of analyses in progress is greater than the local constant MAXANL, or

2) After adding the required number of double-buffers for the new analysis, the total number of 2 x 64-word buffers in the 1K RAM exceeds the local constant NBUF.

Return 2 shall be performed if the SC bit in the AUXMT entry is set for either the prime EFN or one of the suspected contemporaneous EFN's. Otherwise, Return 3 shall be performed to return control to the calling routine.

3.1.3.2 Initiate Analysis Routine (ABINT) -

ABINT shall be called to perform all necessary initiation for an analysis. ABINT shall receive a pointer to an analysis queue entry. The analysis module code shall be retrieved from the queue entry and used as an index to call the initiator corresponding to the analysis type. After the initiator is called, the analysis queue entry block shall be returned to the EXEC's pool of free message blocks. Control shall then be returned to the calling routine.

3.1.3.2.1 Scan Analysis Initiator (ABSAI) - ABSAI shall perform all necessary initialization for scan analysis. (This routine has the capability of initializing any analysis type. As such, it can be used as the initiator for any new analysis types added to the ABI Management 2 Functional Group (e.g., contemporaneous, PRI, etc.)). It should be noted that ABSAI uses the "Start" flag of the AMT entry to cause the Time Out Check Routine (ABTCK) to actually enable the buffers in the 1K RAM. Thus, the analysis would actually start on the next ABI Management 2 Tick (50 ms). It would be possible to start an analysis immediately by substituting the start-analysis-instruction-sequence in ABTCK for the setting of the start flag.

ABSAI shall receive a pointer to an analysis queue entry in the X-register. Subroutine ABAMT shall then be called to find a non-valid entry in the Analysis Managment Table. ABAMT shall return the AMT entry address in the B-register. The AMT entry shall be initialized as follows:

1) Word 0: Valid flag is set by ABAMT. Start flag set by ABSAI.
2) Words 1-8: Data from analysis start message.
3) AAT Pointers: Assigned by ABASN (Assign ABI Buffers Routine)
4) Word 10: Set by ABTCK (Time Out Check)
5) Word 11: 0 (Not used in scan analysis)
6) Word 12, 13: 0
7) CA Counters: 0
8) Word 15: Set by ABREQ (Output Aux Bus Request messages)

3.1.3.2.1.1 Assign AMT Storage (ABAMT)

ABAMT shall search the Analysis Management Table (AMT) for a non-valid entry. Prior testing performed in ABRCK shall guarantee the existence of an available entry. When the vacant entry is found, the local variable NANAL (Number of Analyses in Progress) shall be incremented and the valid flag in the entry set. A pointer to the newly assigned AMT entry shall be returned to the calling routine in the B-register.

3.1.3.2.1.2 Output Aux Bus Request Messages (ABREQ)

ABREQ shall output all Aux Bus Request messages destined for the CP which are required for an analysis. A pointer to the AMT entry for the analysis to be serviced shall be passed to ABREQ in the B-register. ABREQ shall also compute the value of TTAMP and store it in the AMT entry. The calculation algorithm:

1) If scan analysis TTAMP = (EFPAMP - X'A')/2
2) Otherwise TTAMP = MINAMP (a local constant)

ABREQ shall call ABRQ1 to actually format and send the Aux Bus Request message to the CP via EXMSG. ABRQ1 shall receive the EFN in the A-register and TTAMP in the E-register.

3.1.3.2.1.3 ABI Buffer Control Routines

There shall be four routines to control the usage of the eight 2 x 64 word double-buffers in the 1K RAM. The routines shall be:

1) ABASN, which shall assign buffers, to an analysis,
2) ABREL, which shall release buffers from an analysis,
3) ABINB, which shall enable the depositing of SPDW's for an EFN in a buffer, and

4) ABPAU, which shall inhibit the depositing of SPDW's for an EFN in a buffer.

These routines shall all receive the same input, namely a pointer to the AMT entry in the B-register. They shall be analysis-oriented and can be used to assign buffers for any type of analysis, even multi-double-buffered contemporaneous analysis.

3.1.3.2.1.3.1 Assign ABI Buffers (ABASN) - ABASN shall link Analysis Buffer Assignment Table (AAT) entry pairs to an AMT entry. The AAT shall be searched for a non-valid entry pair. The pointer to this AAT entry pair shall be saved in the AMT. The AAT entry pair shall be initialized as follows:

1) Primary entry:

Word Ø: Valid flag set
Primary flag set
EFN field set to EFN from AMT

Word 1: Pointer to AMT entry

2) Secondary entry:

Word Ø: EFN field set to EFN from AMT

Word 1: Pointer to AMT entry

It should be noted that 1, 2, 3, or 4 AAT entry pairs can be linked to an AMT entry. AAT entry pairs shall consist of a primary and secondary entry, each entry representing a single 64 word buffer in the 1K RAM.

3.1.3.2.1.3.2 Release ABI Buffers (ABREL) - ABREL shall clear the valid bit of all AAT entry pairs linked to the AMT entry.

3.1.3.2.1.3.3 Initialize ABI Buffers (ABINB) - For each AAT entry pair assigned to the AMT entry, ABINB shall set the primary entry loading flag and format and set the primary entry buffer control word to indicate:

1) Buffer Interrupt enabled
2) PDW Pointer = 0
3) Buffer status = Buffer empty
4) Track number = EFN (from AAT sub-entry)

3.1.3.2.1.3.4 Paulse ABI Buffers (ABPAU) - For each AAT entry pair assigned to the AMT entry, ABPAU shall set the buffer control word corresponding to both the primary and secondary AAT entry to indicate "buffer not valid".

3.1.3.3 Analysis Purging

3.1.3.3.1 Purge Test 1 (ABPG1) - ABPG1 shall attempt to purge an analysis in order to free buffers in the 1K RAM or possibly free AMT storage. ABPG1 shall require no register input from the calling routine. It shall search the AMT for a valid low priority entry (priority = 2 or 3). If such an entry exists, ABPG1 shall call Analysis Purge (ABPRG) to actually purge the analysis and shall then execute the second return to the calling routine (Purge performed). If no such AMT entry exists, the first return shall be executed (Cannot purge). This return would be executed if the resources were allocated to high priority analyses.

3.1.3.3.2 Purge Test 2 (ABPG2) - ABPG2 shall attempt to purge an analysis in order to free an EFN that is already under analysis. ABPG2 shall receive the EFN in the A-register. It shall search the AMT for a valid low priority entry (priority = 2 or 3) which is processing

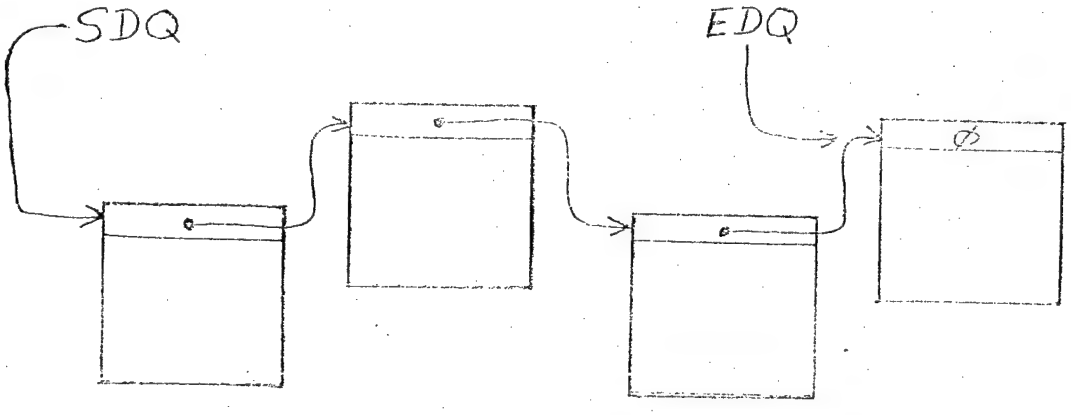
SPDW's for that EFN, either as the prime EFN or a suspected contemporaneous EFN. If such an entry exists, ABPG2 shall call Analysis Purge (ABPRG) to actually purge the analysis and shall then execute the second return to the calling routine (Purge performed). If no such AMT entry exists, the first return shall be executed (Cannot purge).

3.1.3.3.3 Analysis Purge Routine (ABPRG) - ABPRG shall purge the analysis, whose AMT entry address is passed to ABPRG in the B-register. The purge sequence shall consist of:

1) Releasing ABI buffers (call to ABREL).
2) Returning data blocks to free block queue (call to ABRTB).
3) Outputting Aux Bus Control messages to stop SPDW's for all EFN's associated with this analysis (call to ABSTP)
4) Outputting a Null Analysis Return message via EXMSG.
5) Clearing the valid flag in the AMT entry.

3.1.3.3.3.1 Data Block Management

ABGTB (Get Block) and ABRTB (Return Block) shall be the two routines performing the data blocks management function. ABGTB shall get one data block (Four 16-bit words) from a queue of "free" blocks. ABRTB shall return one or more data blocks to the free block queue. The free block queue shall be defined by a start of queue pointer (SDQ) and an end of data queue pointer (EDQ). The queue and blocks shall have the following structure:



3.1.3.3.3.1 Data Block Management - continued -

SDQ shall be used in getting blocks. EDQ shall be used in returning blocks. If several blocks are returned via one call to ABRTB they themselves shall be a linked chain of blocks.

3.1.3.3.3.2 Output Aux Bus Stop Messages (ABSTP)

ABSTP shall output all Aux Bus Control messages to stop SPDW's for all EFN's associated with an analysis. The register input to ABSTP shall be the address of an AMT entry in the B-register. The SPDW Stop shall be formatted and output via EXMSG for the primary EFN. If the analysis is contemporaneous, stop messages shall be formatted and output for the suspected contemporaneous EFN's.

3.1.4 ABI Return Processing

3.1.4.1 ABFUL - Buffer Full Processing -

ABFUL shall be called by the AP EXEC each time a buffer full interrupt is received. ABFUL shall read the contents of the ABI data buffer full status register at the time of the interrupt. (See ABI Design Specification document). The contents shall be saved in a Start-ABRDR message buffer. ABSWP shall then be called for each buffer that is flagged as full in order to enable the co-buffer, i.e., to perform buffer swapping. ABSWP shall be passed the address of an AAT entry in the B-register. After all required buffer swapping has been completed, the Start-ABRDR message shall be output via EXMSG. Control shall then be returned to the AP Exec's buffer full interrupt handler.

3.1.4.2 ABI Return Driver -

ABRDR shall be called by the AP Exec to process the data in one or more of the sixteen 64-word buffers in the 1K RAM. ABRDR

shall receive from the AP Exec the address of a Start-ABRDR message in the X-register. The contents of the buffer full status register shall be retrieved from the message and the message block returned to the AP Exec's queue of free message transfer blocks. For each buffer whose status is full, the AMT entry address shall be retrieved from the corresponding AAT entry. The analysis module code shall be retrieved from the AMT entry and used as an index to call the appropriate data accumulation routine for this type of analysis. If the accumulation routine returns via the "aborted" return (return no. 1), the search for buffers flagged as full shall continue. If the accumulation routine returns via the "completed" return (return no. 2), the count of buffers to be processed has been exceeded, i.e., sufficient data has been gathered to allow a computation. This type of analysis completion has not been implemented for scan analysis but may be desirable in other types. If the data accumulation has been completed, the "done" flag shall be set in the AMT entry, the ABI Buffers paused (ABPAU) and a Start-ABDDR message sent to the EXEC. If the accumulation routine returns via the normal return (return no. 3), the search for buffers flagged as full shall continue. After all buffers whose bit in the data buffer full status register was set have been processed, control shall be returned to the AP EXEC.

3.1.4.3 Swap ABI Buffers (ABSWP) -

ABSWP shall be called to enable the other half of an ABI double-buffer when the currently active half has been filled. ABSWP shall receive in the B-register the address of the AAT entry whose corresponding buffer in the IK RAM has been filled. The full flag shall be set in that AAT entry and the second entry in the AAT entry pair shall be located. In the second entry, the loading flag shall be set. Buffer control for the buffer control word assigned to the second entry shall be set as follows:

(1) Buffer interrupt enabled

(2) PDW pointer = Ø

(3) Status = buffer empty

(4) Track number = EFN from AAT sub-entry.

Control shall then be returned to the calling routine.

3.1.4.4 Scan Data Accumulation (ABSAC) -

ABSAC shall process a full or partially full buffer in the 1K RAM. ABSAC shall receive a pointer to the AAT entry, whose data buffer is to be processed. ABSAC shall have three subroutine returns:

(1) Analysis had to be purged. There is an insufficient quantity of data blocks in the free data block queue to process this buffer.

(2) Analysis has been completed. This return would not normally apply to an accumulation routine for a scan-like analysis (i.e., for a timed analysis, the only condition leading to completion is "maximum time exceeded"). However, ABSAC shall use this return if the free data block queue has been exhausted and no purging of any analysis can be accomplished.

(3) Normal return. Processing of the buffer data has been accomplished.

ABSAC shall first determine the number of PDW's present in the buffer (ABSAC may be called to process a partially full buffer after the scan time-out has been exceeded). For each valid PDW then, one half of the PDW pulse amplitude (PAMP) shall be compared against value TTAMP. (TTAMP shall be available in the AMT entry). This test is necessary only because of the interaction of SPDW requests from the RMP and the AP. RMP-sourced requests shall have priority and may require a lower value of TTAMP than AP-sourced requests.

If the pulse amplitude exceeds the threshold, a data block shall be requested from the free data block queue via a call to ABGTB and the block attached to the data queue of the AMT entry. For scan analysis, the PDW time of arrival (both MSB's and LSB's) and the pulse amplitude shall be saved in the block. This process shall be repeated until all PDW's in the block have been processed.

If ABGTB (Get Data Block) performs a no-more-blocks return to ABSAC, an attempt shall be made to free up data blocks by purging a low priority analysis (ABPG1). If no purging is possible, the elapsed time of the analysis shall be computed. If the elapsed time exceeds a program constant, the analysis shall be termed "abnormally completed". As such, the abnormal completion bit shall be set in the AMT entry and return no. 2 (analysis completed) shall be performed. If the elapsed time does not exceed the constant, the analysis shall be purged and return no. 1 (analysis purged) shall be performed. If purging has been accomplished, a test shall be made to see if the current analysis has caused itself to be purged. If so, return no. 1 (analysis purged) shall be performed. If not, a second attempt shall be made to get a data block.

3.1.5 ABI Time Out Check (ABTCK)

ABTCK shall be called periodically by the AP EXEC (every 50 ms) to perform two functions:

(1) Initiate any time critical analyses that have been set up.

(2) Check each analysis in progress to see if the time out period for that analysis has been exceeded.

If there are any analyses to be initiated (AMT entries with start flag set), the start flag shall be cleared, the analysis start time

recorded in the AMT entry, and the ABI buffers enabled via a call to ABINB. If any analyses have timed out, the ABI buffers shall be disabled via a call to ABPAU, the done flag in the AMT entry set, and a start-ABDDR message formatted and output via EXMSG. The message shall contain the AMT entry address of the analysis timing out.

3.1.6 ABI Done Driver (ABDDR)

ABDDR shall be called to process all accumulated data for an analysis that has timed out or has met its data quantity requirement. ABDDR shall receive a pointer to a Start-ABDDR message in the X-register. The AMT entry address of the analysis to be processed shall be retrieved from the message and ABDDR shall check the done flag in the AMT entry. If it is not set, the analysis must have been purged before the accumulated data could be processed and a Start-ABIDR message shall be output via EXMSG and control returned to the AP EXEC. This action will result in a search of the analysis queues for new analyses to be initiated. The analysis module code shall then be retrieved from the AMT entry and used as an index to call the proper "analysis done" processing for this analysis type. If the analysis done processing terminates normally, the data blocks from the AMT entry data queue shall be returned to the free data block queue (via a call to ABRTB). Then the Aux Bus control messages to stop SPDW's shall be output (via a call to ABSTP), the flags in the AMT entry cleared, and the local constant NANAL (no. of analyses in progress) decremented. Then the Start-ABIDR message shall be output. If the analysis done processing terminates abnormally, the only action shall be to output the Start-ABIDR message. (This would be the case if no more data blocks were available and there was a partially full buffer to be added to the accumulated data).

3.1.6.1 Scan Analysis Done Processing (ABSDN) -

ABSDN shall perform processing of accumulated data for scan analyses. ABSDN shall receive a pointer to the AMT entry in the B-register. ABSDN shall have two returns available:

(1) Analysis aborted. No data blocks available to allow completion of data accumulation.

(2) Normal. Scan analysis terminated normally.

ABSDN shall first determine if there is any data in a partially full buffer which must be added to the accumulated data before computations can commence. If so, ABSAC shall be called to complete the accumulation. If ABSAC terminates normally, the ABI buffers shall be released (call to ABREL) and the scan type computed (call to ABSTY). The scan type shall then be stored in an analysis return message and the message output via EXMSG. Control shall be returned to the calling routine via the normal return. If ABSAC returns via the aborted return, control shall simply be returned to the calling routine via the analysis aborted return.

3.1.6.1.1 Scan Type Calculation (ABSTY) - ABSTY shall determine the scan type based on the accumulated data in the AMT entry data queue. ABSTY shall receive the address of the AMT entry in the B-register and shall return the scan type in the 4 MSB's of the A-register (Ø's in the LSB's).

ABSTY shall test the AMT entry end-of-data-queue pointer to determine if any pulses have been received. If no pulses were received, the scan type shall be set to sidelobe and control returned to the calling routine. If pulses have been received, ABSTY shall determine the maximum value of the amplitude field of the entries in the AMT entry data

queue. If the maximum value of the amplitude field is less than the value of ATC (Amplitude Threshold Constant - see CDBDD), ABSTY shall set the scan type to sidelobe and shall return to the calling routine. If the maximum received amplitude is greater than ATC, ABSTY shall shift the histogram down in value by the minimum value of the amplitude field so that the first non-zero bin occupied by the histogram is assigned zero amplitude value. The histogram shall be limited at the high end to contain only ten bins numbered 0 to 9.

ABSTY shall then calculate the mean value, $\overline{A}$, of the amplitude according to the formula:

$$\overline{A} = 1/N \sum_{i=0}^{m-1} in_i$$

where m = number of histogram bins;

n_i = histogram count for the i th bin, and

$$N = \sum_{i=0}^{m-1} n_i$$

The variance of the histogram shall be calculated according to the formula:

$$\sigma_A^2 = \overline{A^2} - \overline{A}^2$$

where $$\overline{A^2} = 1/N \sum_{i=0}^{m-1} i^2 n_i$$

ABSTY shall calculate the test statistic, t_s, as the ratio of the variance to the mean value, that is:

$$t_s \quad \frac{\sigma_A^2}{9 - \overline{A}}$$

ABSTY shall test t_s against local constants K_1 and K_2 to make the scan type decision as follows:

if $0 \leq t_s \leq K_1$, conclude steady scan.

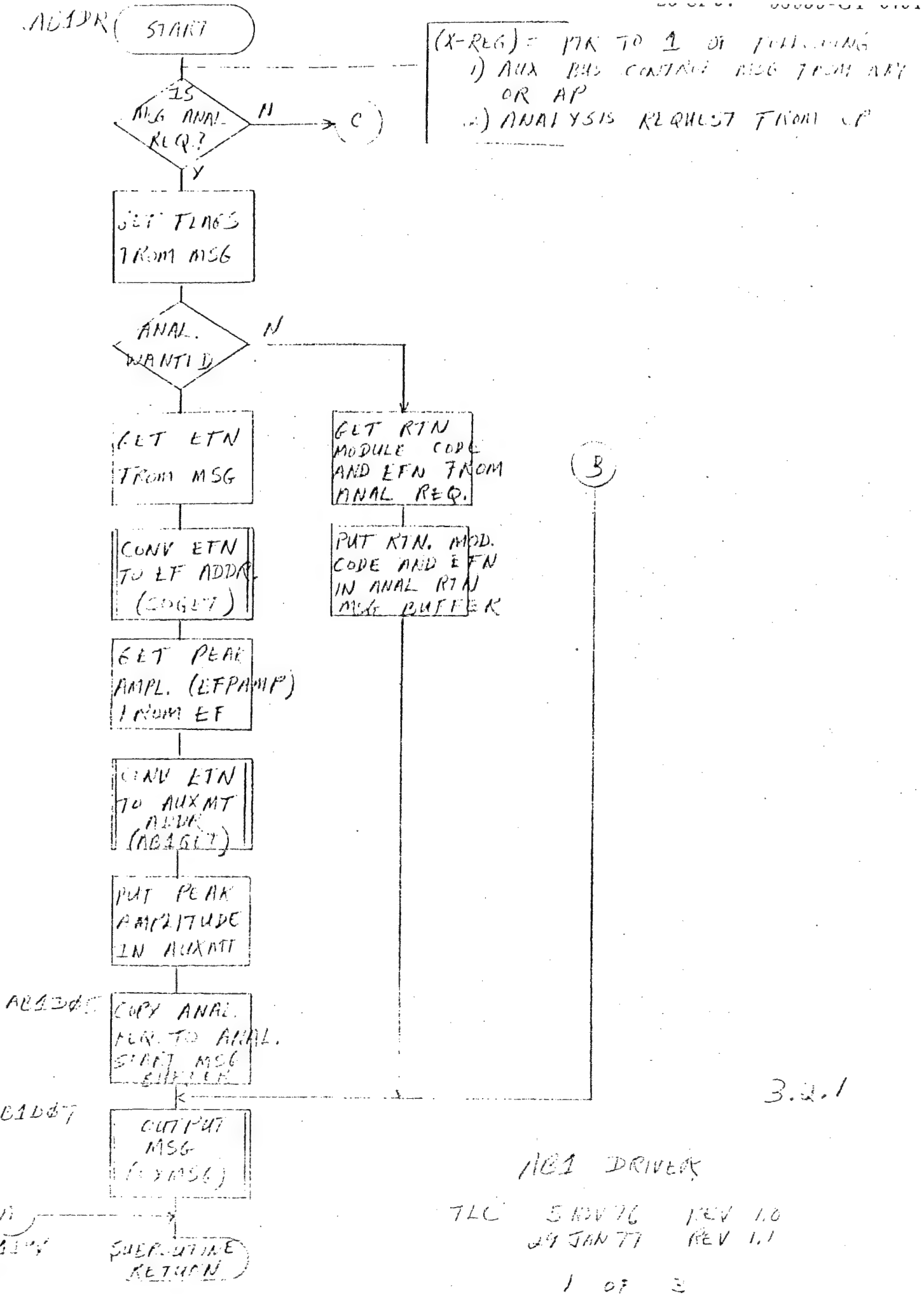
if $K_1 \leq t_s \leq K_2$, conclude conical scan.

if $K_2 < t_s < \infty$, conclude sector scan.

Control shall then be returned to the calling routine.

3.2 SUBPROGRAM FLOW DIAGRAMS

The logic flow for all routines comprising this subprogram is shown in the following flow diagrams. The flow diagrams are labeled so as to correspond to paragraph 3.1. That is, flow diagram 3.2.3 is described in paragraph 3.1.3. Data extraction points for instrumentation are shown as comment blocks with the text "DP____".



3.2.1

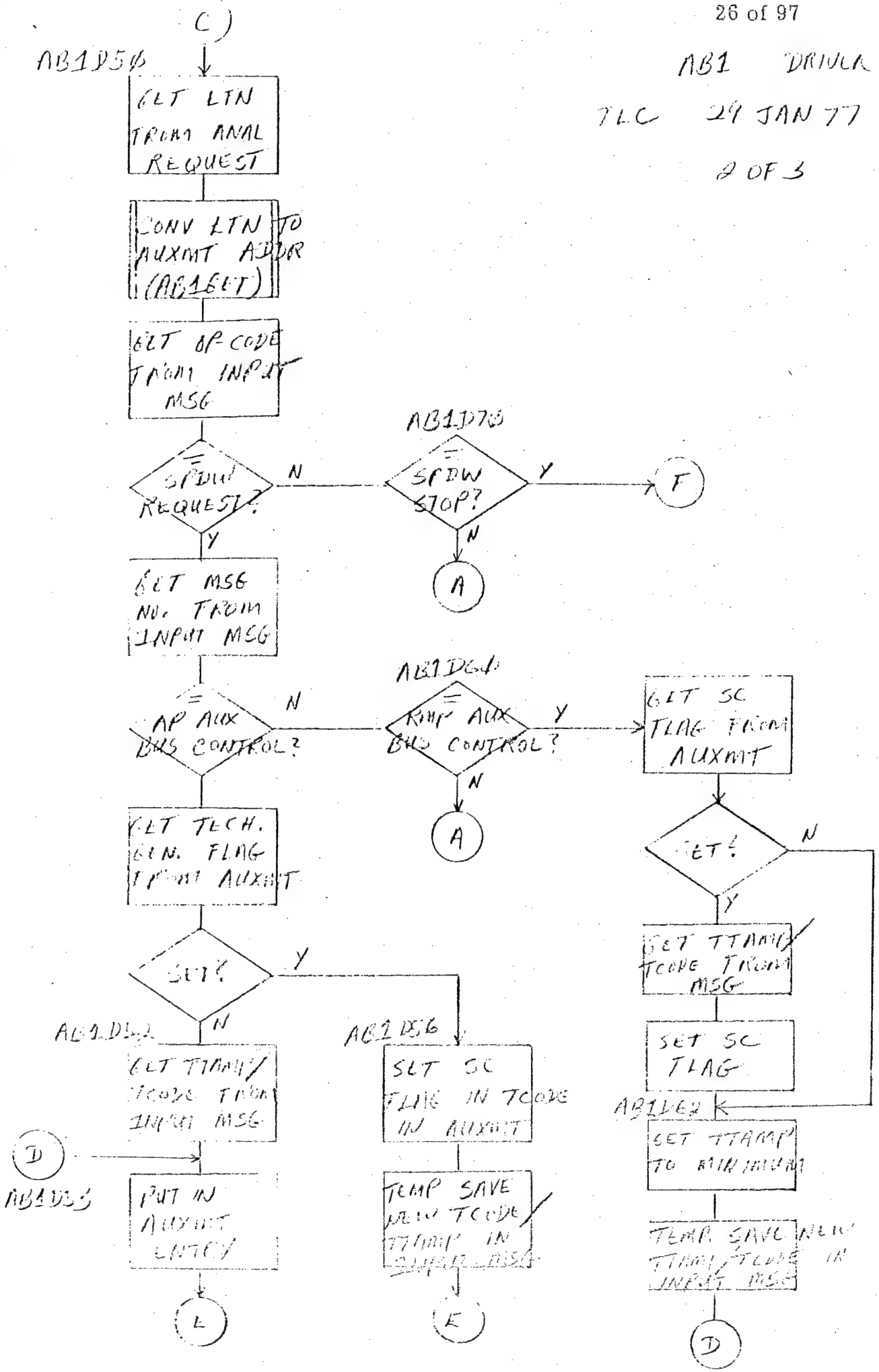
AD1 DRIVER

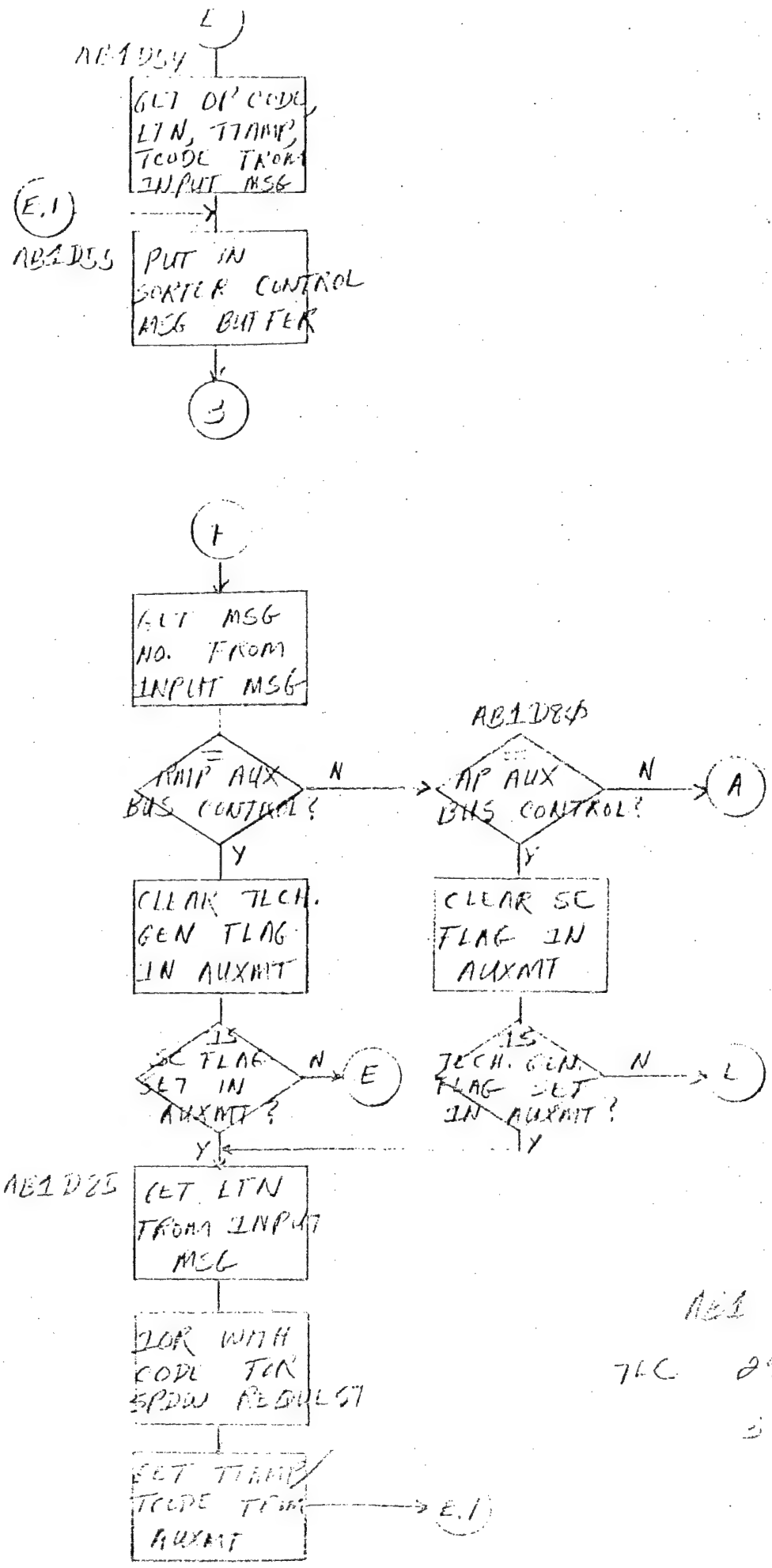
TLC 5 NOV 76 REV 1.0
29 JAN 77 REV 1.1

AB1 DRIVER

TLC 29 JAN 77

2 OF 3

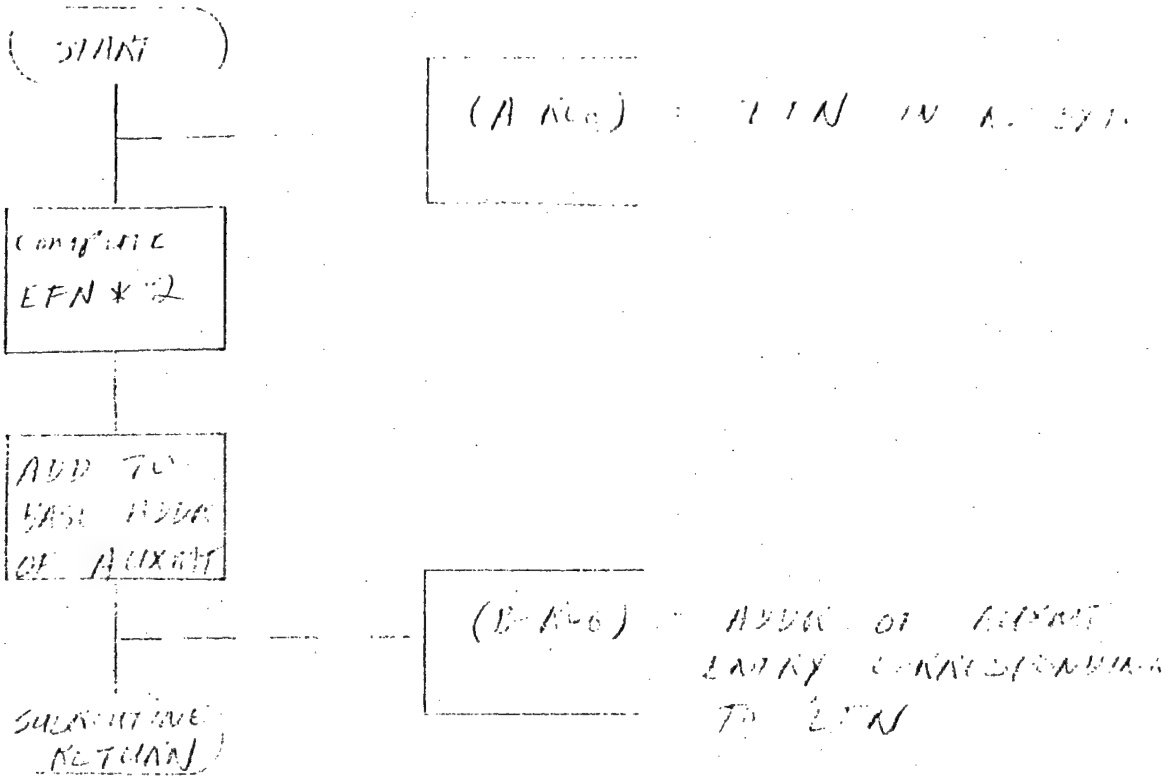




AB1 DRIVER

TLC 29 JAN 77

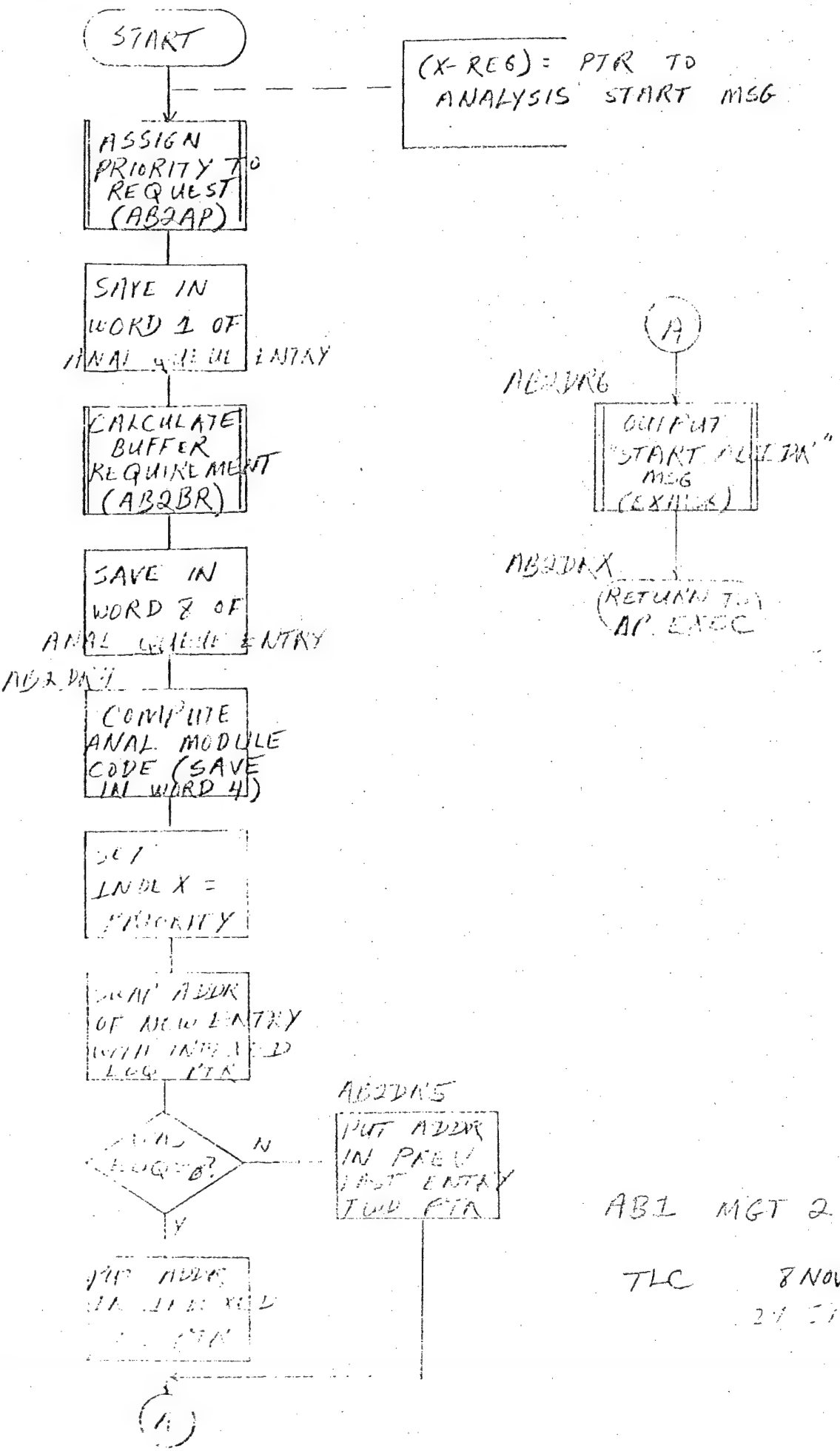
3 of 3



CONVERT EFN TO AUXBIT ENTRY ADDR.

TAC ... JAN 77

AB2DR

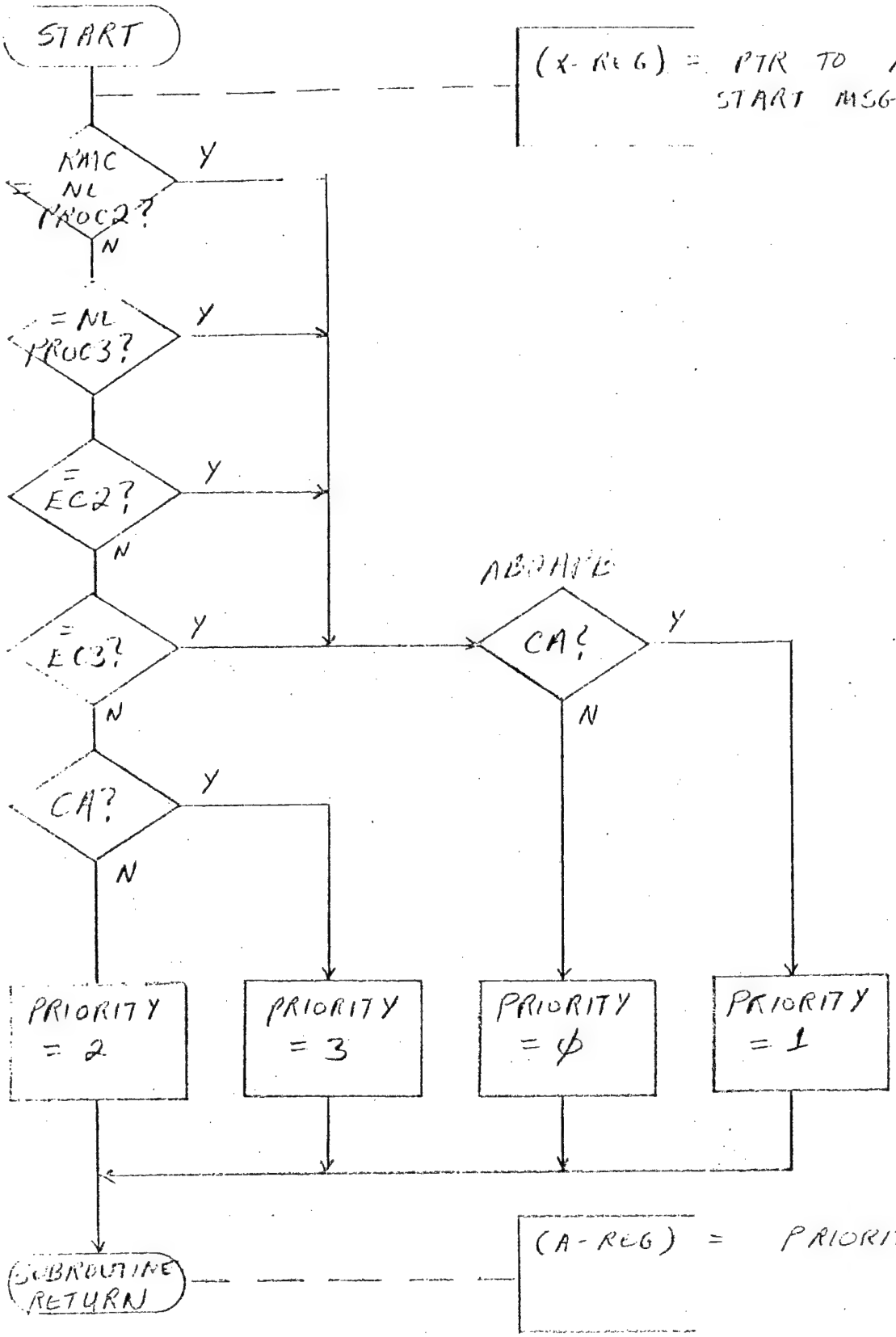


3-2-2

AB1 MGT 2 DRIVER

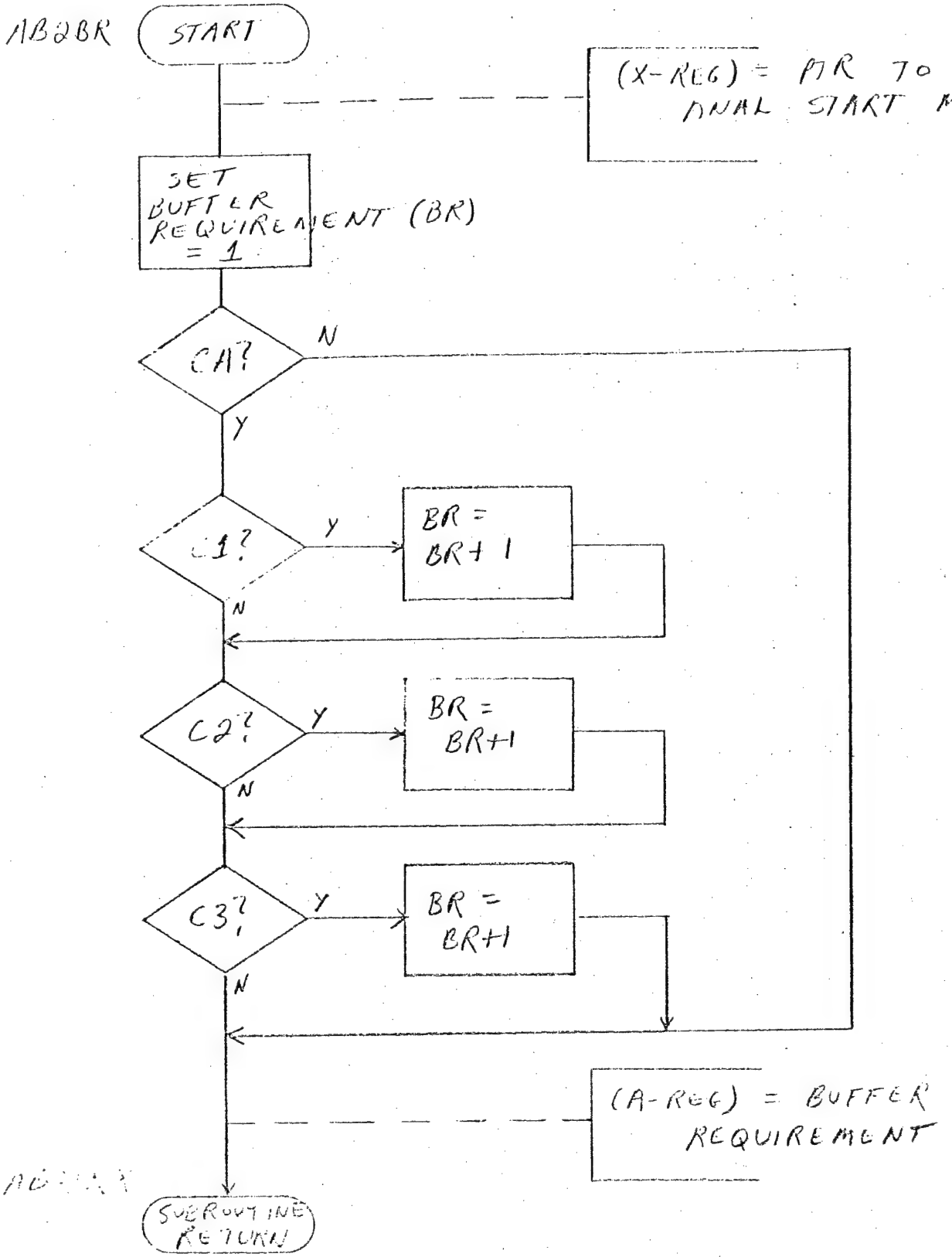
TLC 8 NOV 76

. ABZAP



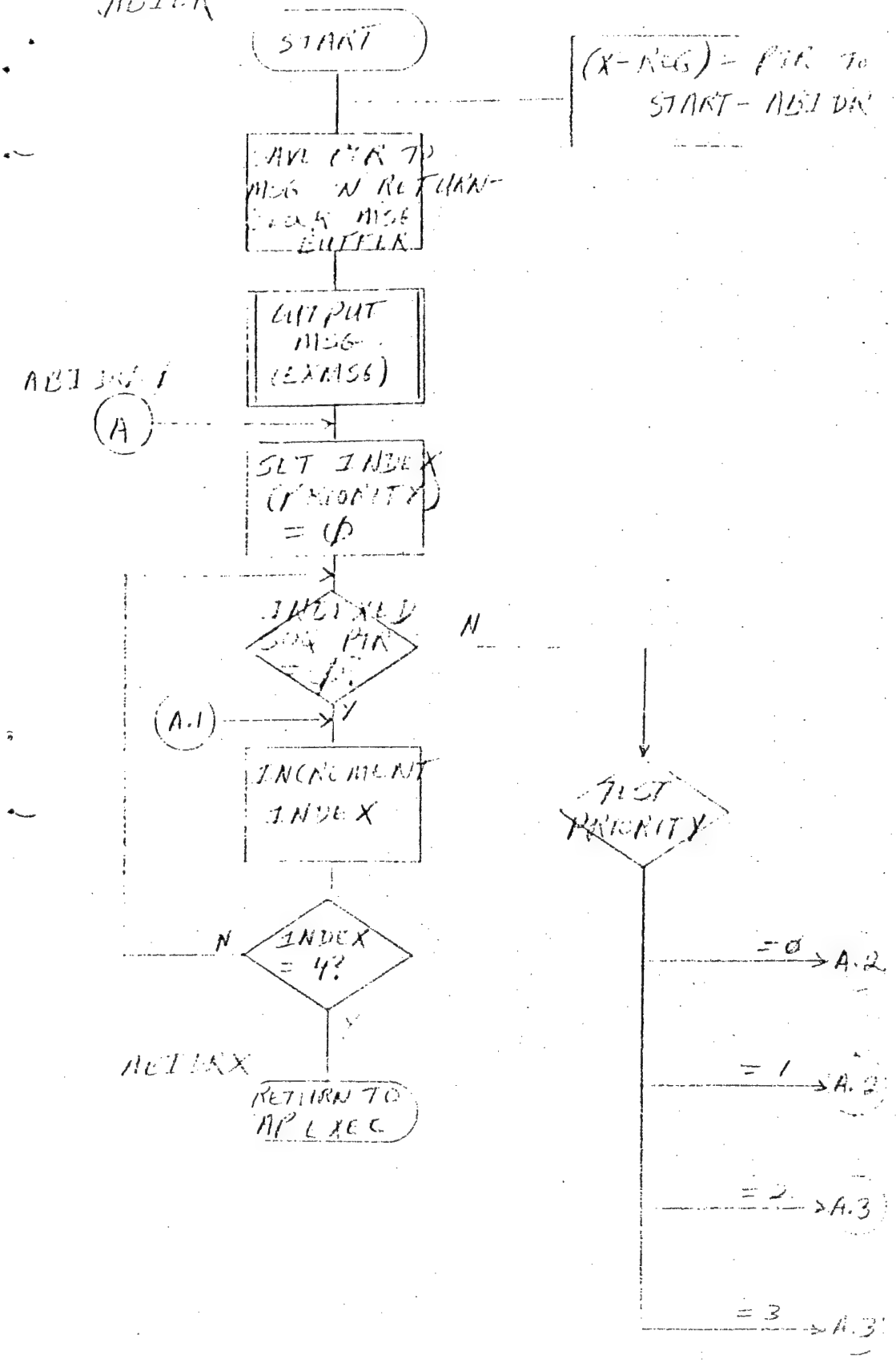
ASSIGN PRIORITY TO ANALYSIS START MSG

TJC 16NOV76



CALCULATE BUFFER REQUIREMENT

TLC 16NOV76

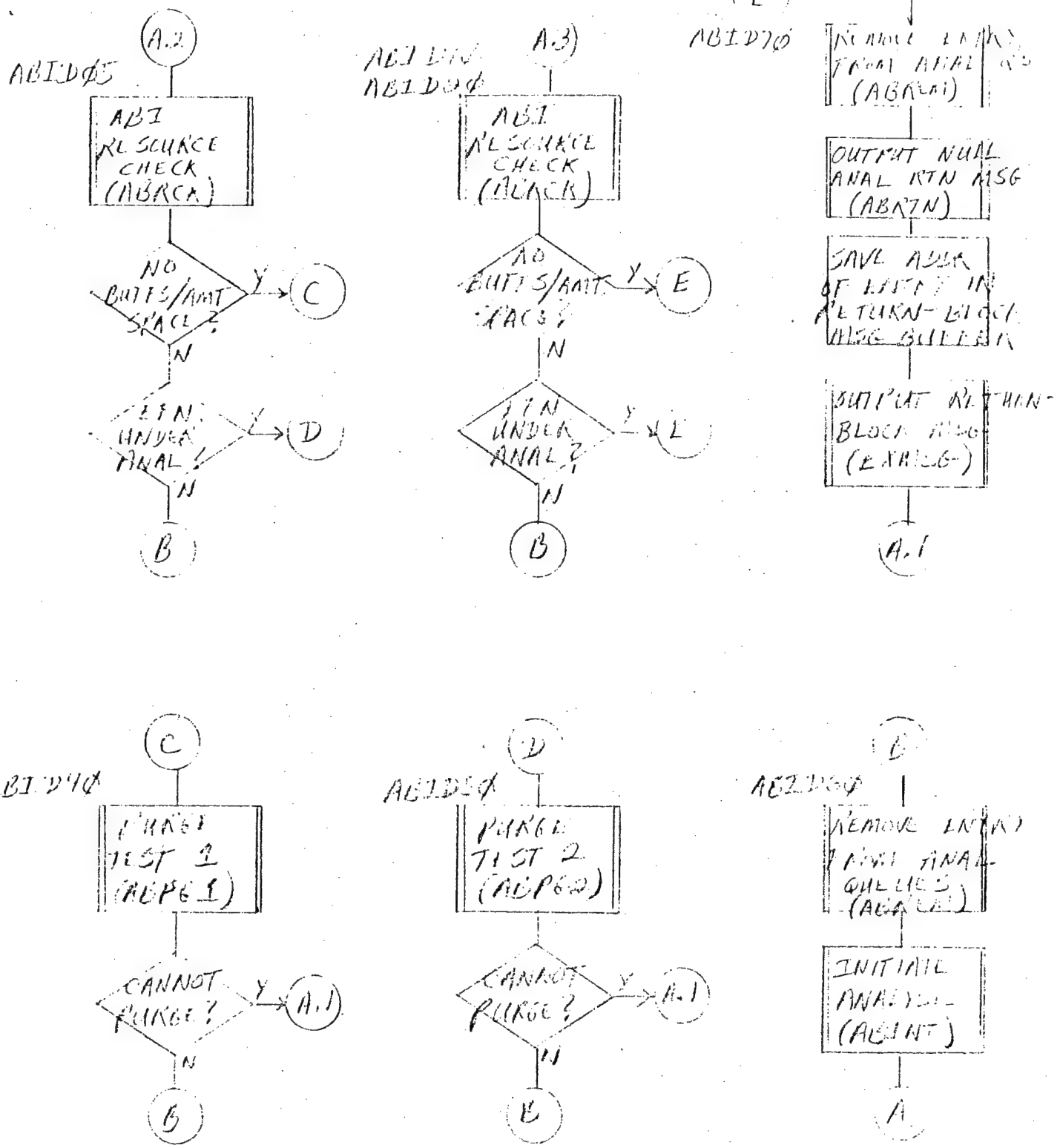


3.2.3

1 of 2

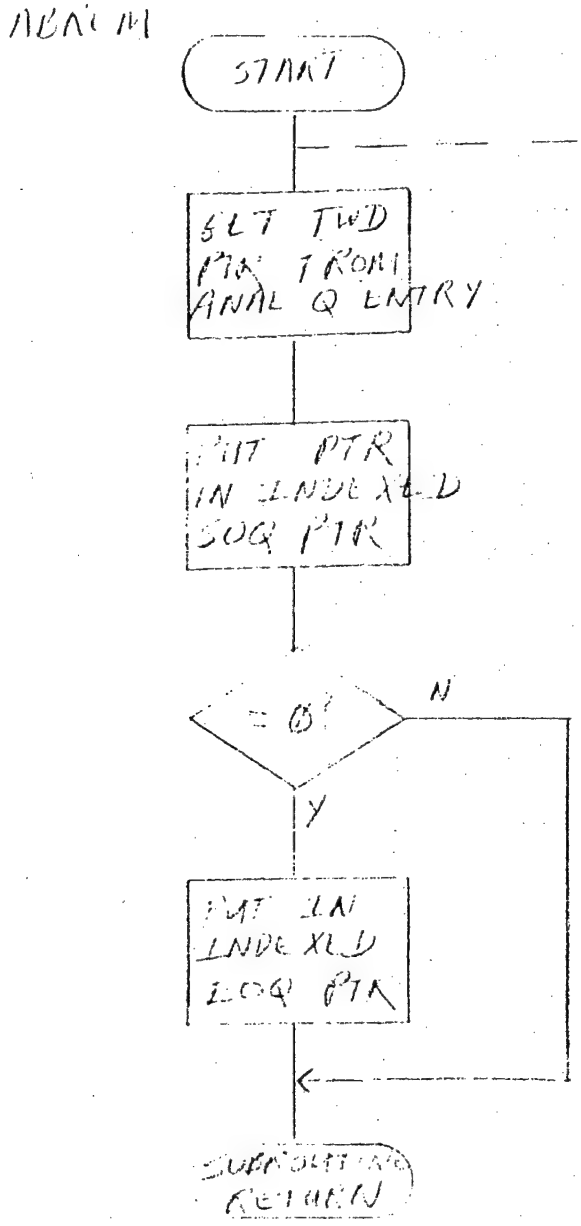
ABI NET.
INITIALIZATION

TLC ENGR'G REV 1.0
REV 1.1



NEI MGT. —
INITIALIZATION DRIVER

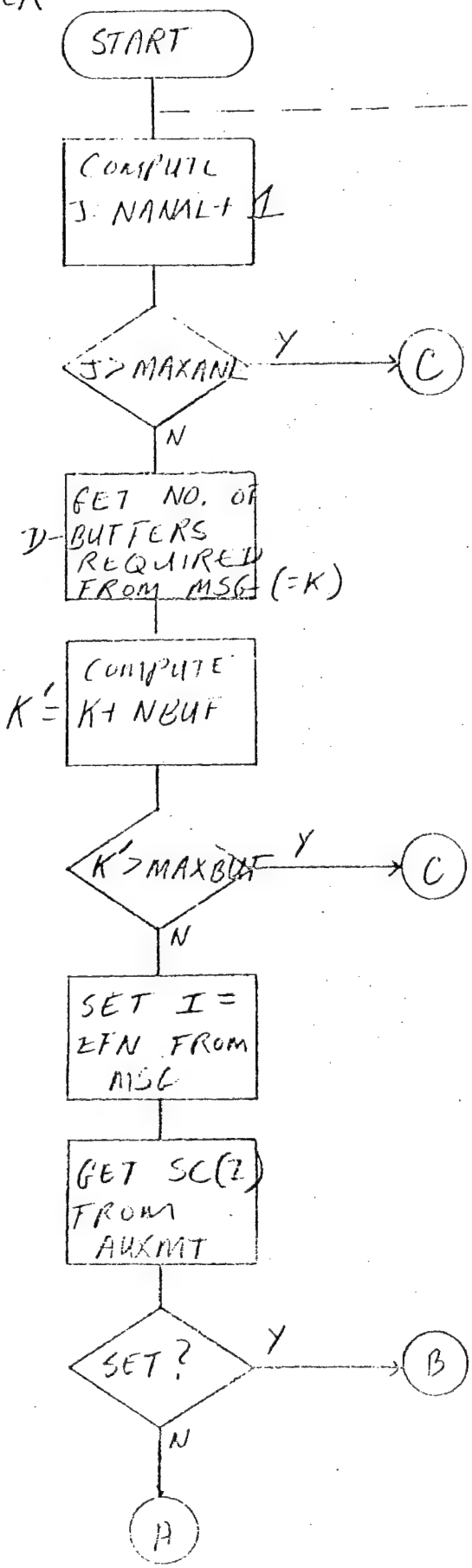
TLC 21 JAN 77



REMOVE ENTRY FROM ANALYSIS QUEUES

TLC 29 JAN 77

ABRCK

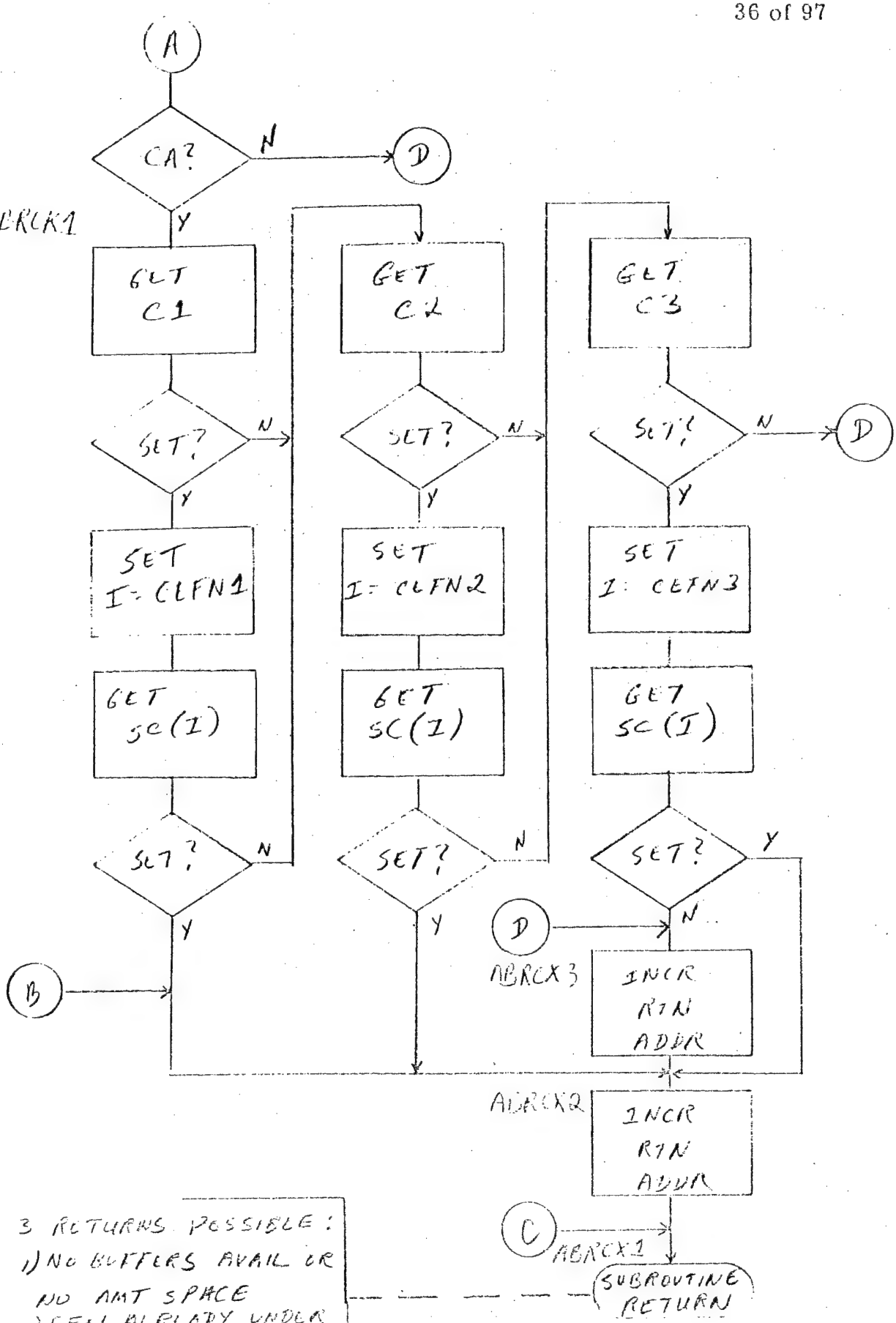


3.2.3.

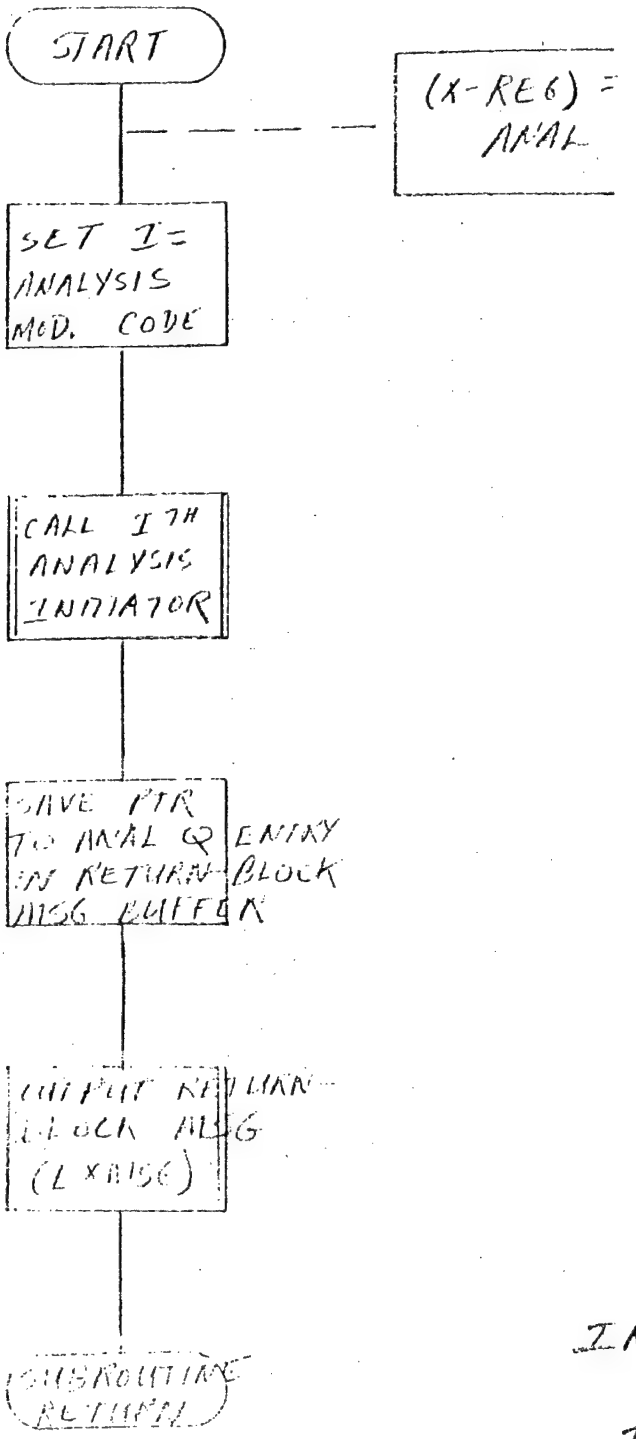
ABI RESOURCE CHECK

TLC 8 NOV 76

1 OF 2



ABI RESOURCE CHECK
TLC 8 NOV 76
2 OF 2

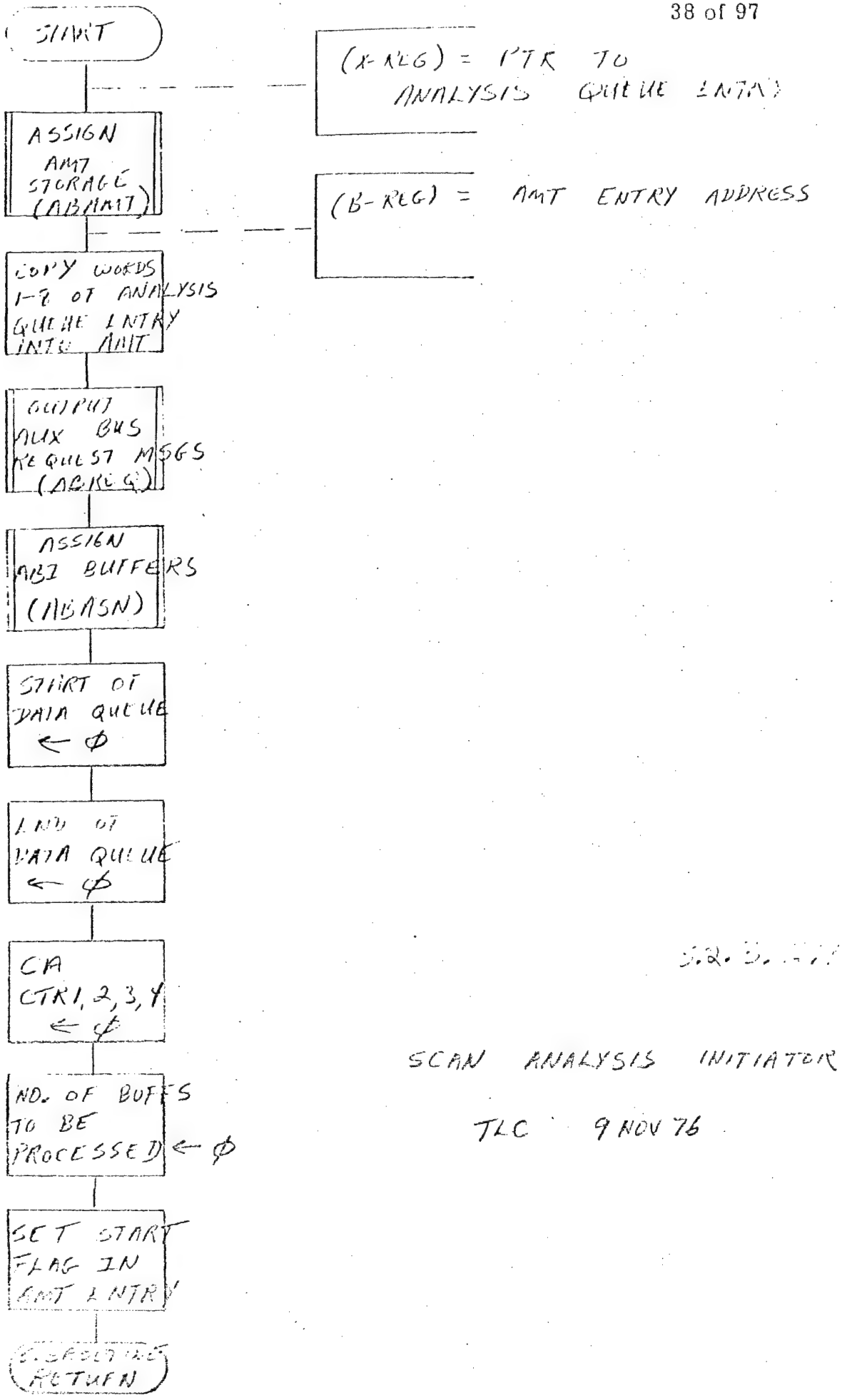


3.2.3.2

INITIATE ANALYSIS

TLC 9 NOV 76 REV 1.0
27 JAN 77 REV 1.1

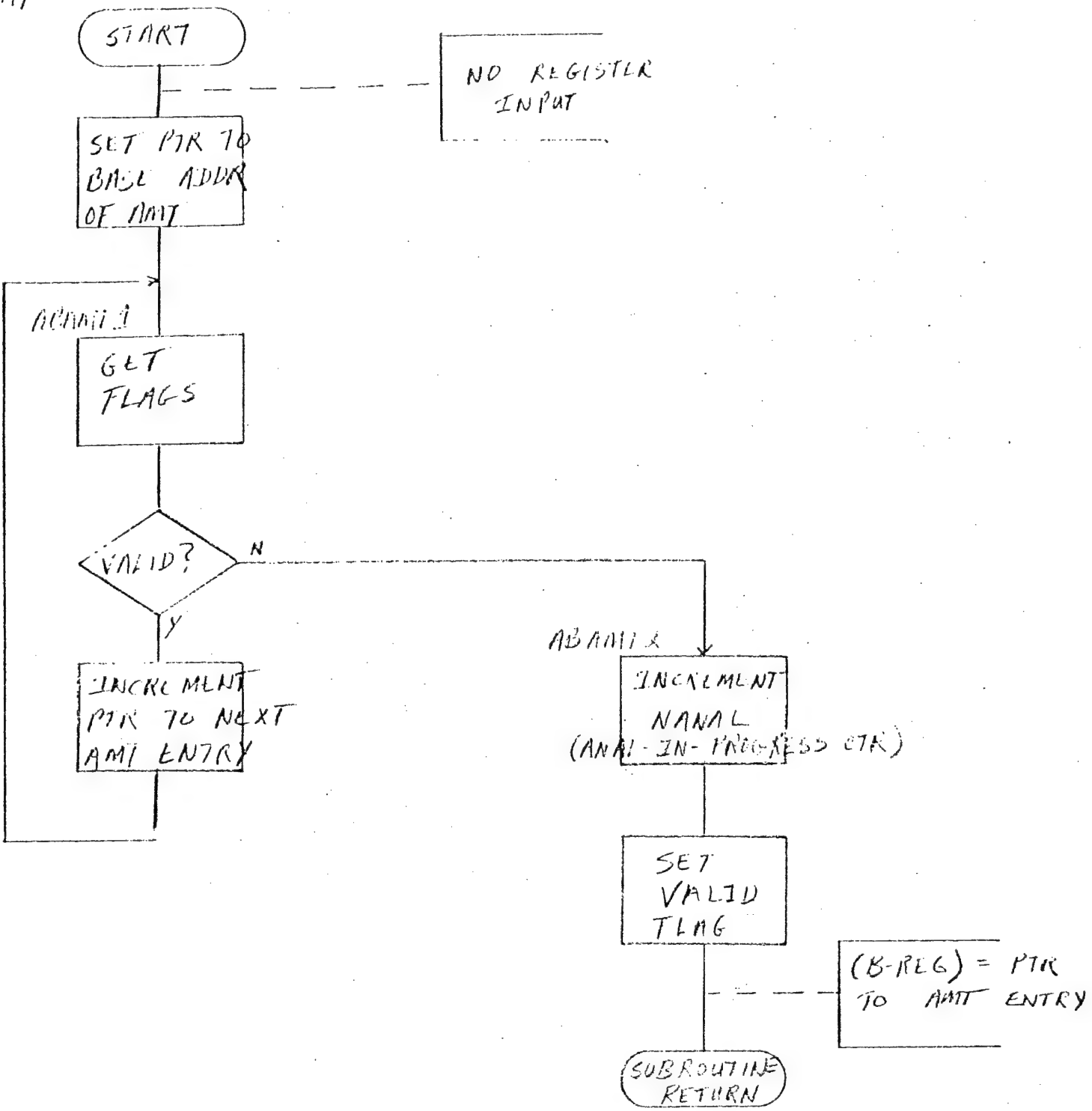
ABSAI



SCAN ANALYSIS INITIATOR

TLC 9 NOV 76

ABAMT

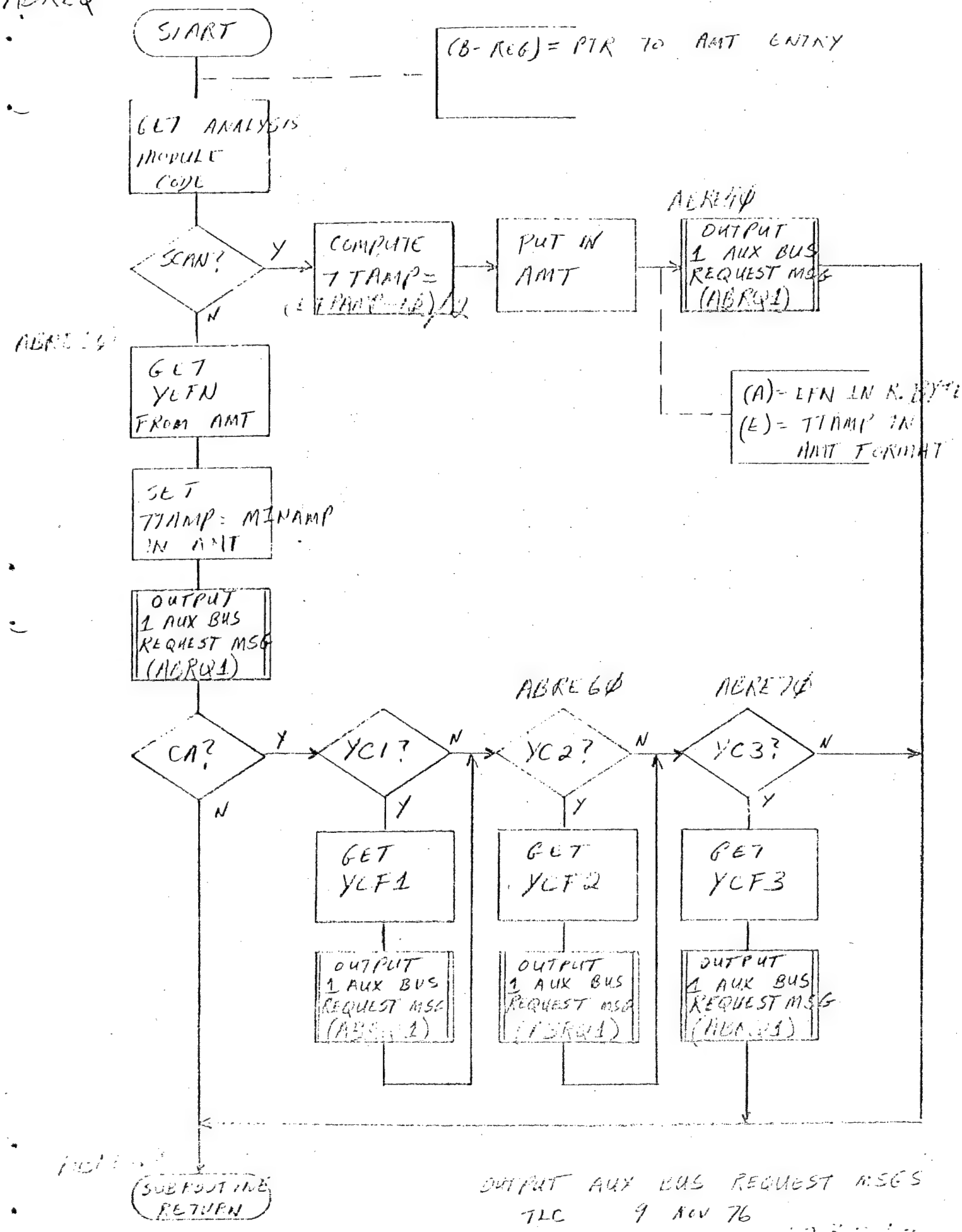


ASSIGN AMT STORAGE

JLC 9NOV76 REV 1.0
29 JAN 77 REV 1.1

3.2.3.2.1.1

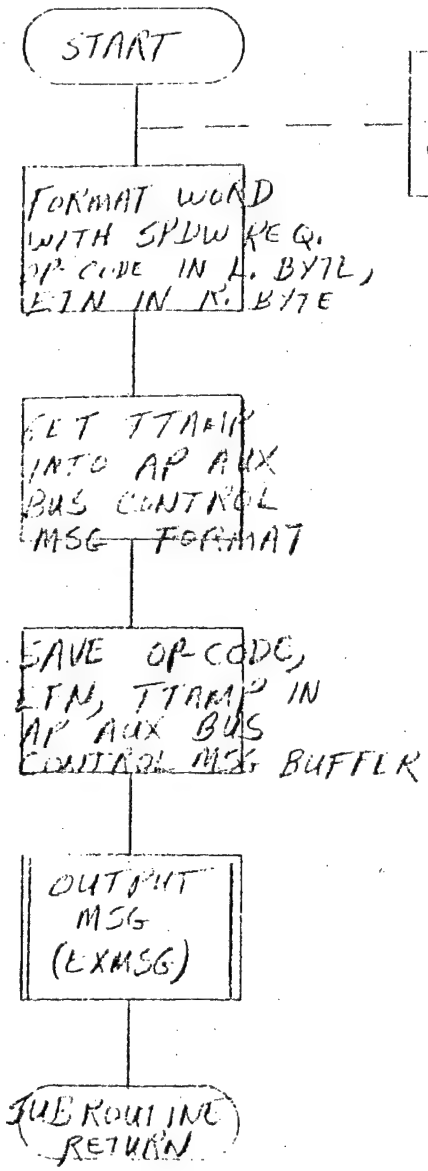
ABREQ



OUTPUT AUX BUS REQUEST MSGS

TLC 9 Nov 76

ABKQ1



(A-REG) = LTN IN R. BYTE
(L-REG) = TTAMP IN AMT FORMAT (BITS 14-11)

OUTPUT 1 AUX BUS REQUEST

TLC 09 JAN 77

ABASN

START

(B-REG) = PTR TO AMT ENTRY

SET N = COUNT OF BUFFERS REQ'D

SET PTR TO START OF AAT

ABASN1

GET FLAGS

VALID?

N

Y

ABASN2

INCR PTR TO NEXT AAT ENTRY PAIR

SAVE AAT ENTRY PAIR ADDR IN AMT

GET LFN FROM AMT

SET VALID, PRIME FLAGS

SAVE FLAGS/LFN IN AAT PRIMARY ENTRY WORD Ø

SAVE LFN ONLY IN AAT 2NDARY ENTRY WORD Ø

INCREMENT DOUBLE-BUFFS IN USE CTR (NBUF)

SAVE PTR TO AAT ENTRY IN WORD Ø OF BOTH PRIMARY AND 2NDARY ENTRIES

DECREMENT N

=Ø?

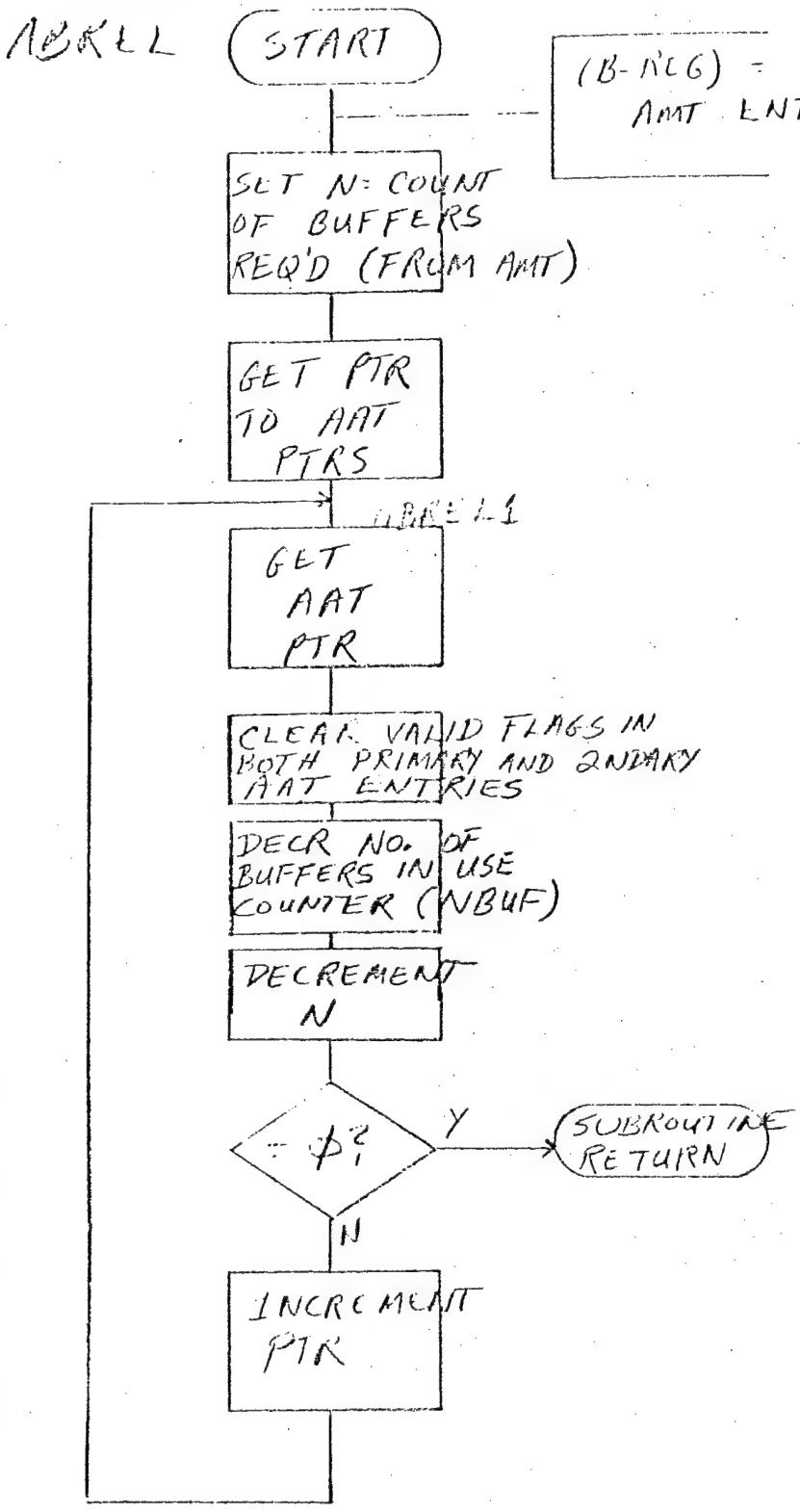
Y

SUBROUTINE RETURN

N

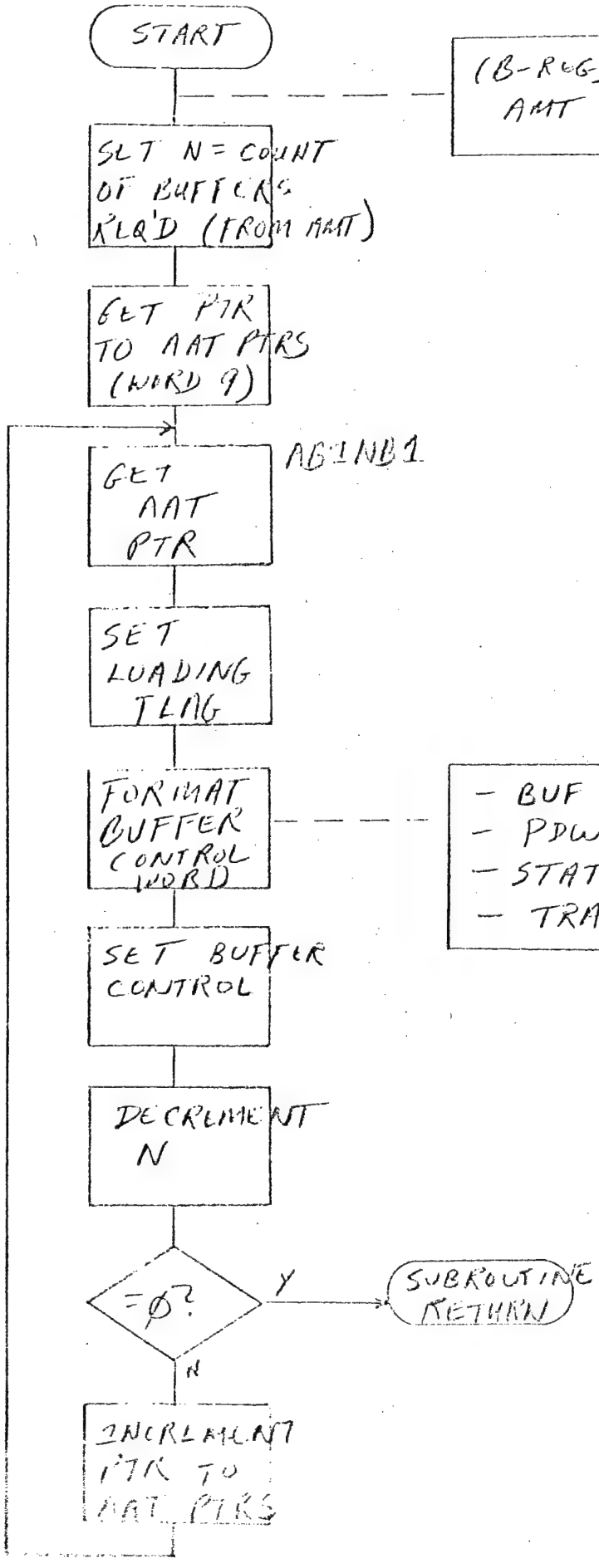
ASSIGN ABI BUFFERS

TLC 8 NOV 76



RELEASE ABI BUFFERS

TLC 8 NOV 76

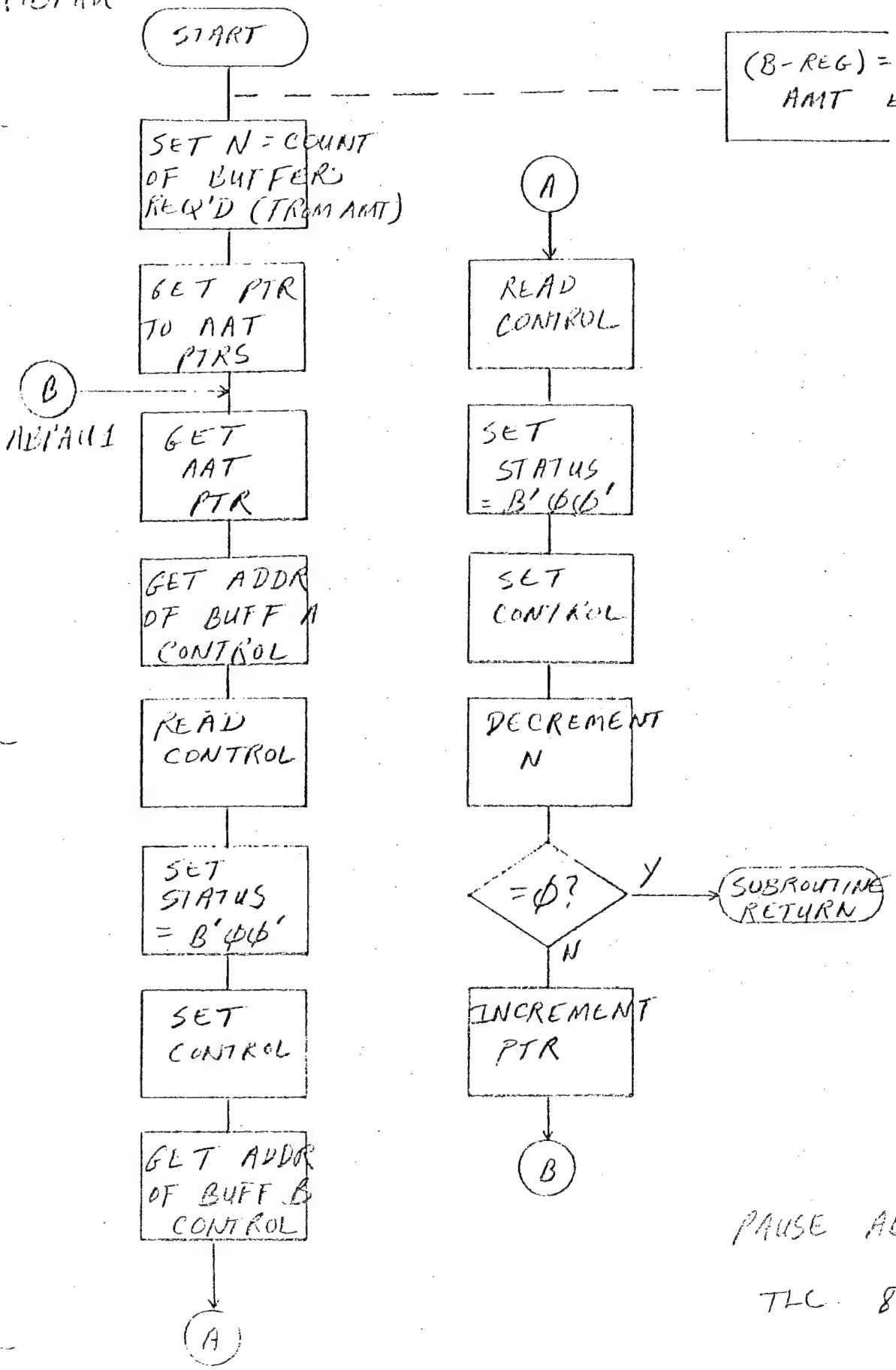


INITIALIZE ABI BUFFERS

TLC 8 NOV 76

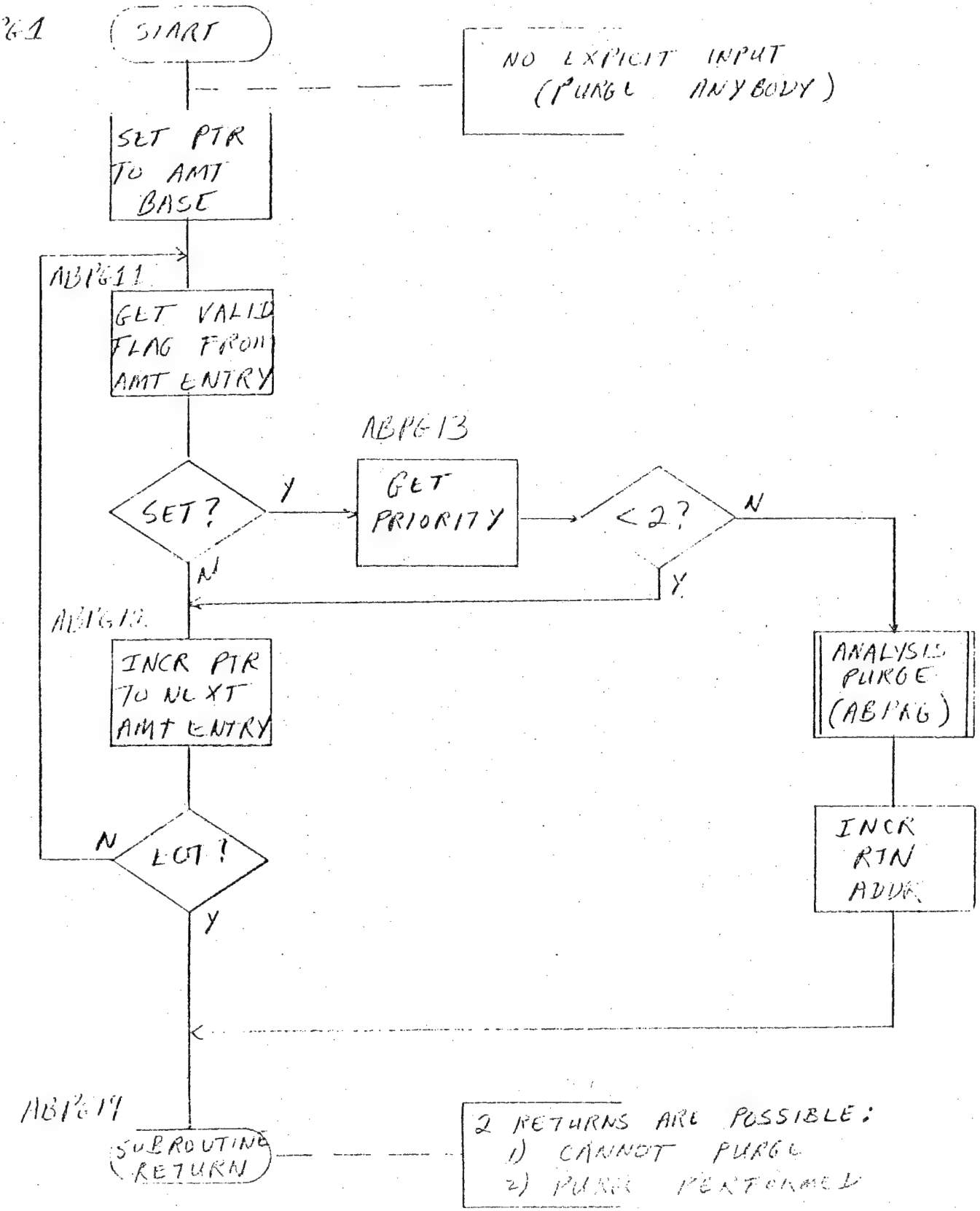
3.2.3.3.1.3.3

ABPAM



PAUSE ABI BUFFERS

TLC 8 NOV 76



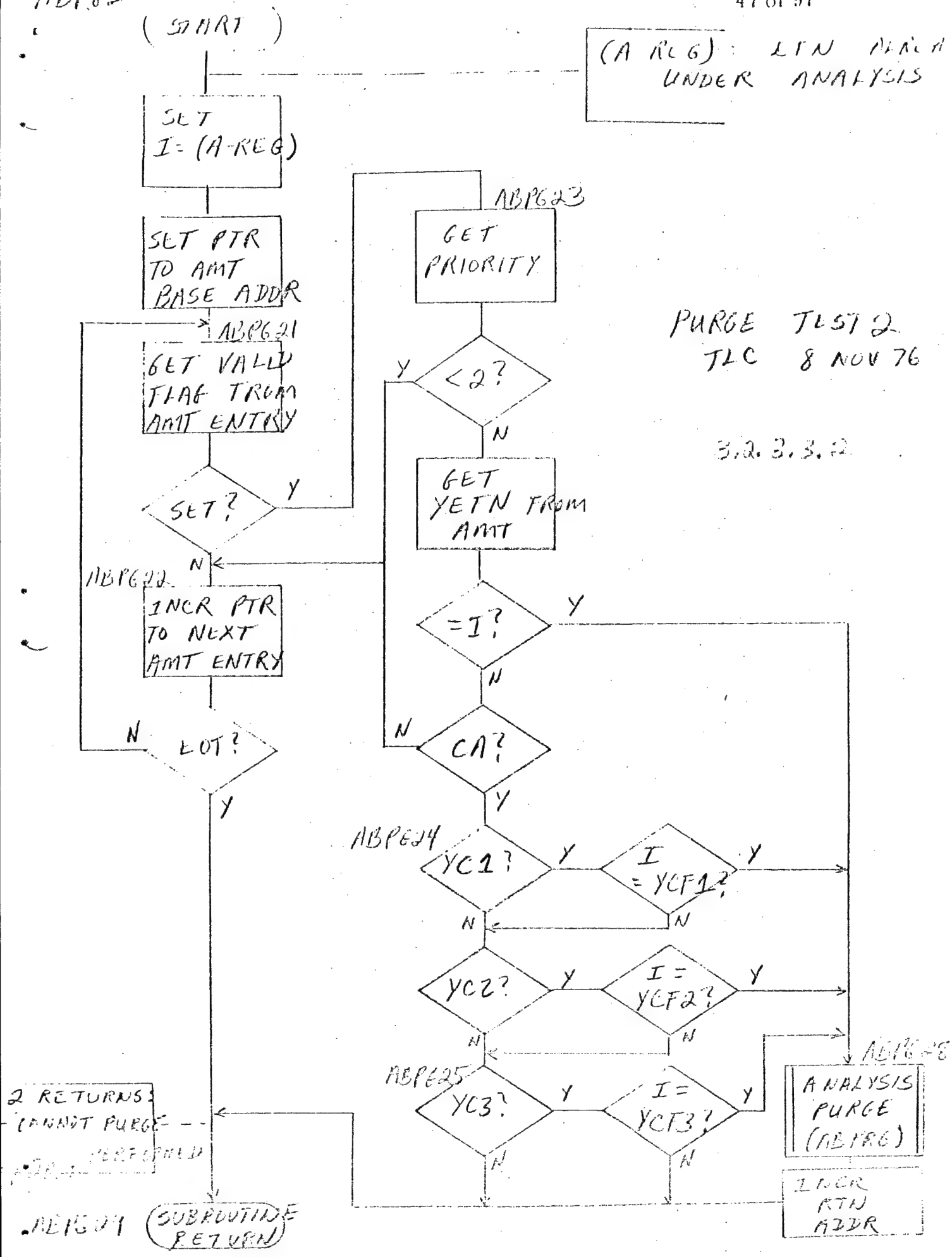
PURGE TEST 1
TLC 8 NOV 76

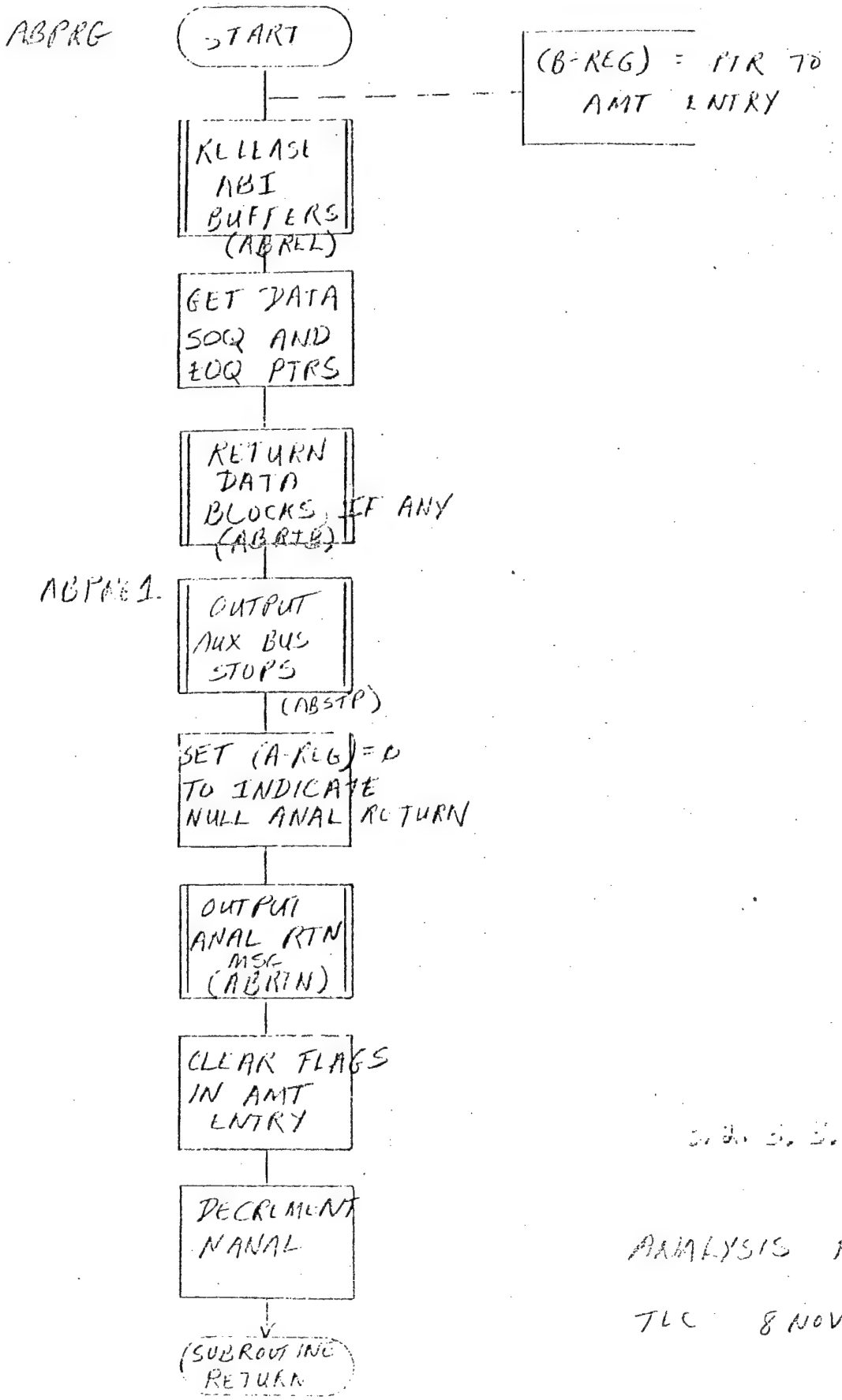
B.B.B.B.!

ABP62

PURGE TEST 2
TLC 8 NOV 76

3.2.3.3.2

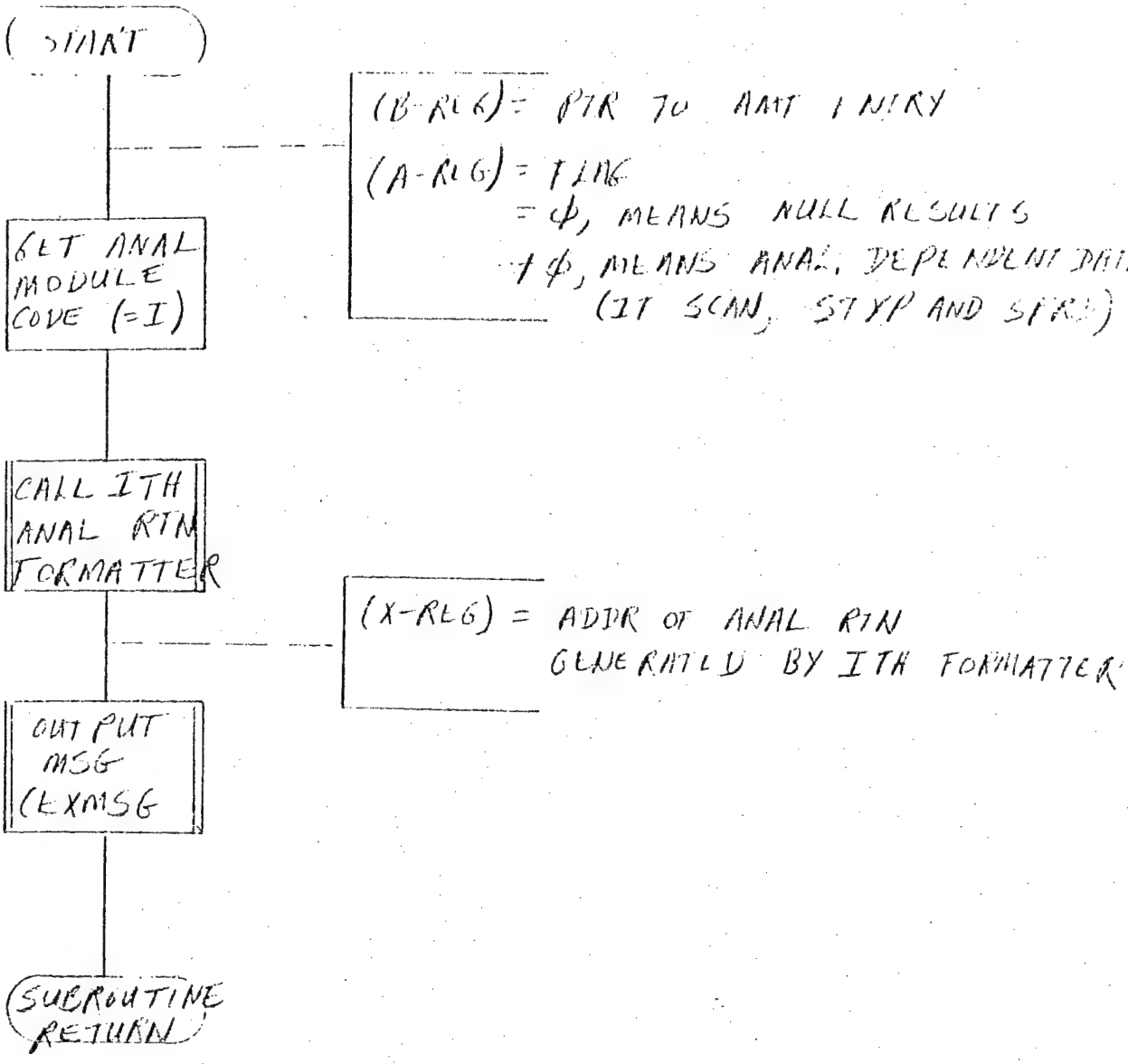




ANALYSIS PURGE

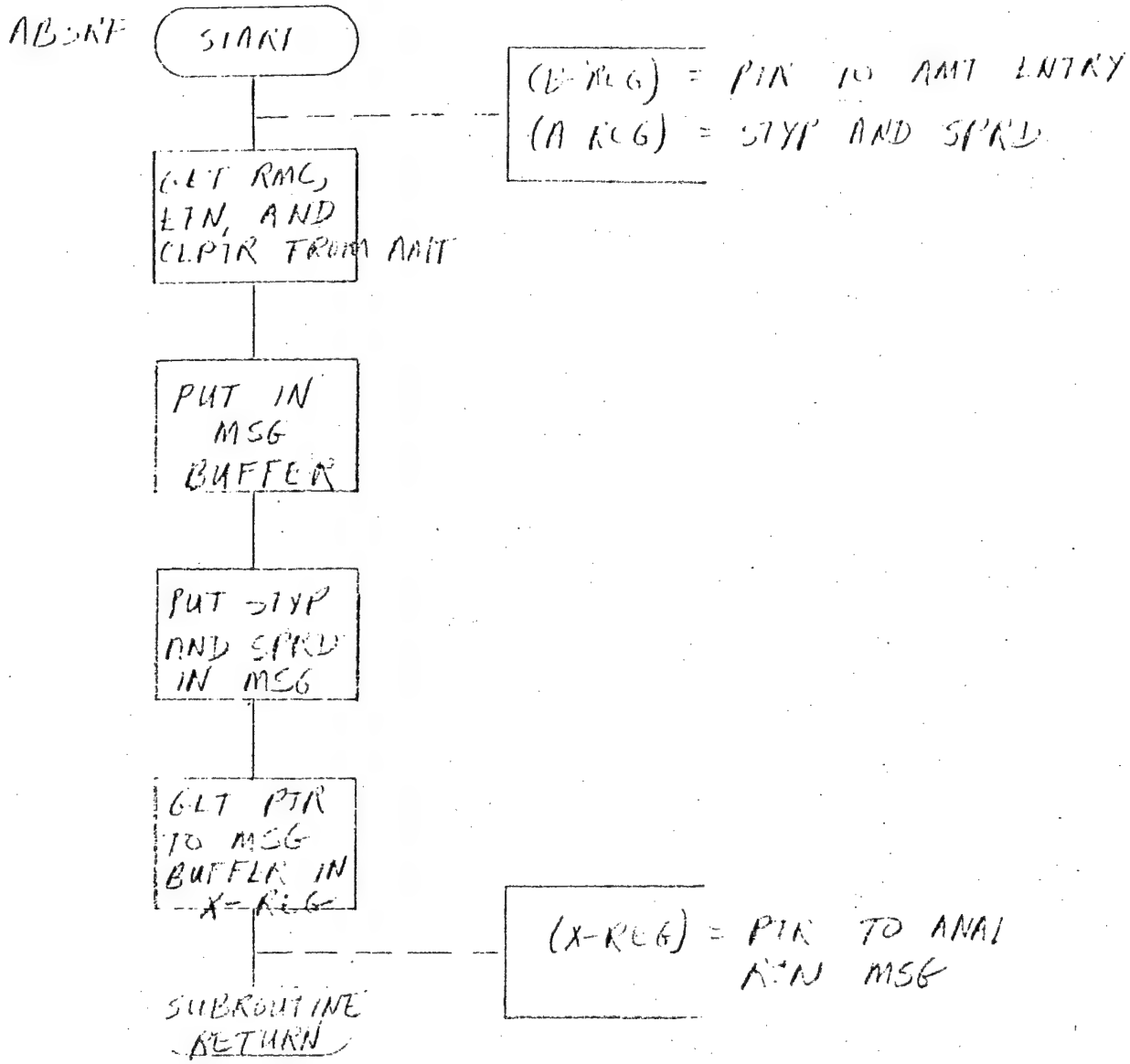
TLC 8 NOV 76

NBRIN



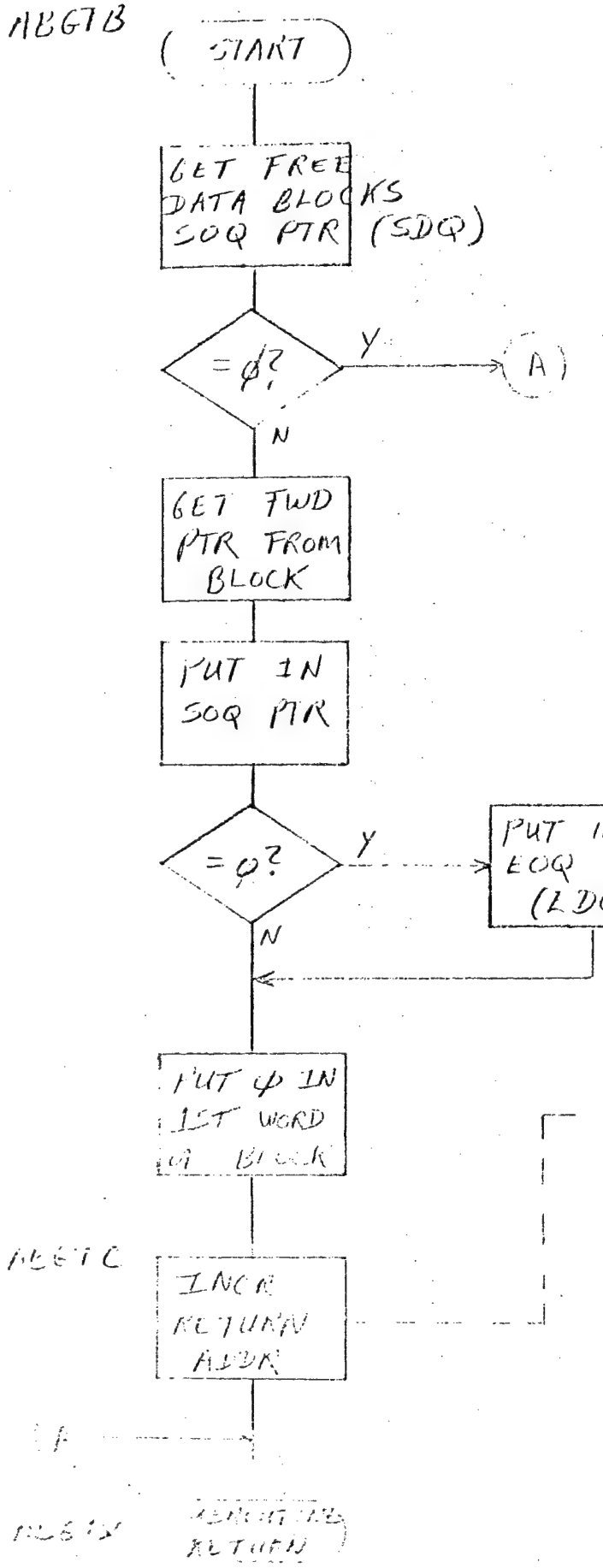
OUTPUT ANALYSIS RETURN MSG

TLC 27 JAN 77



SCAN ANALYSIS RETURN FORMATTER

TLC 3130N77



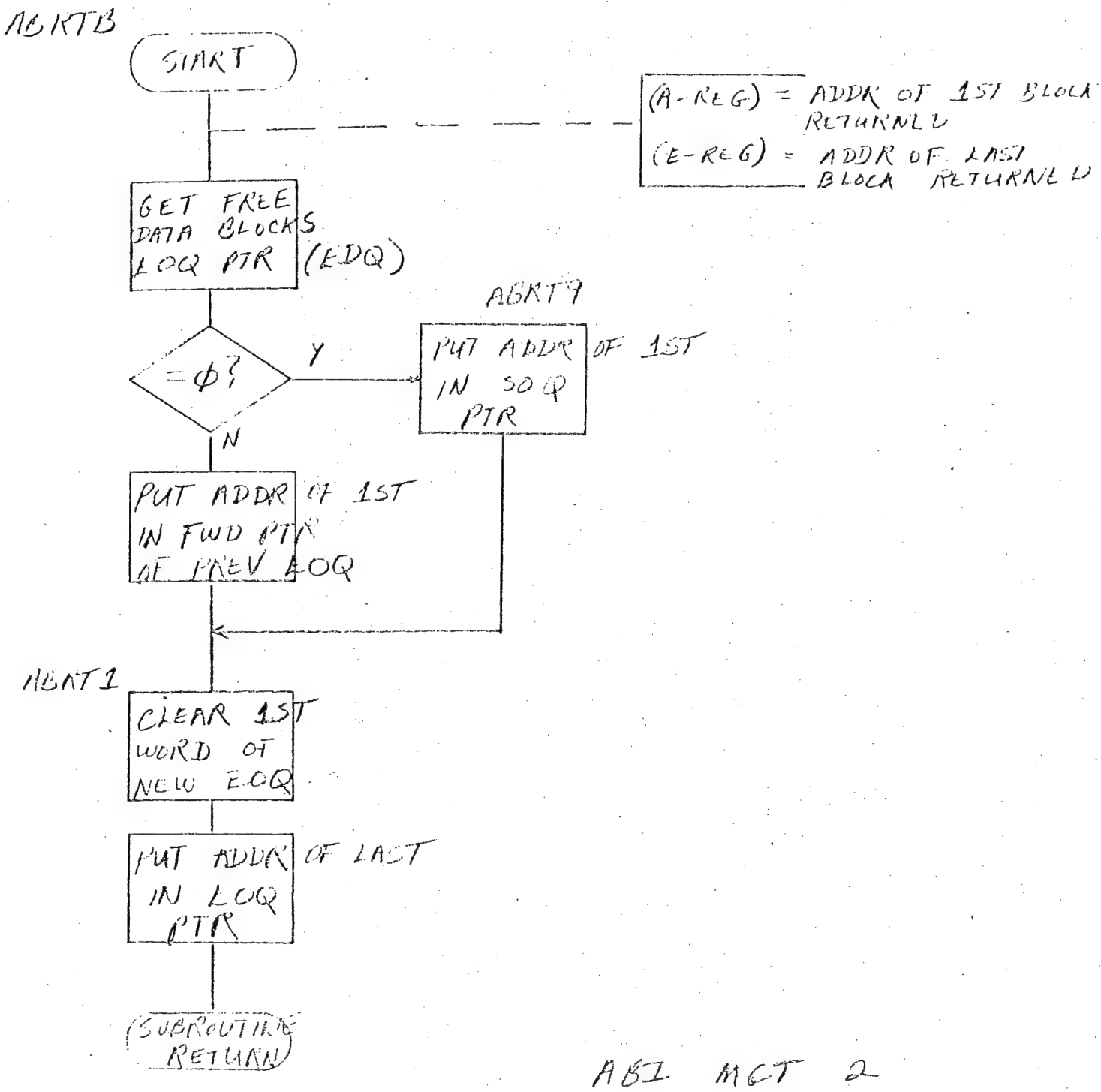
2 RETURNS
1) NO MORE BLOCKS
2) NORMAL. (A-REG) = ADDR OF BLOCK

ABI MGT 2

GET DATA BLOCK

TLC 16 NOV 76

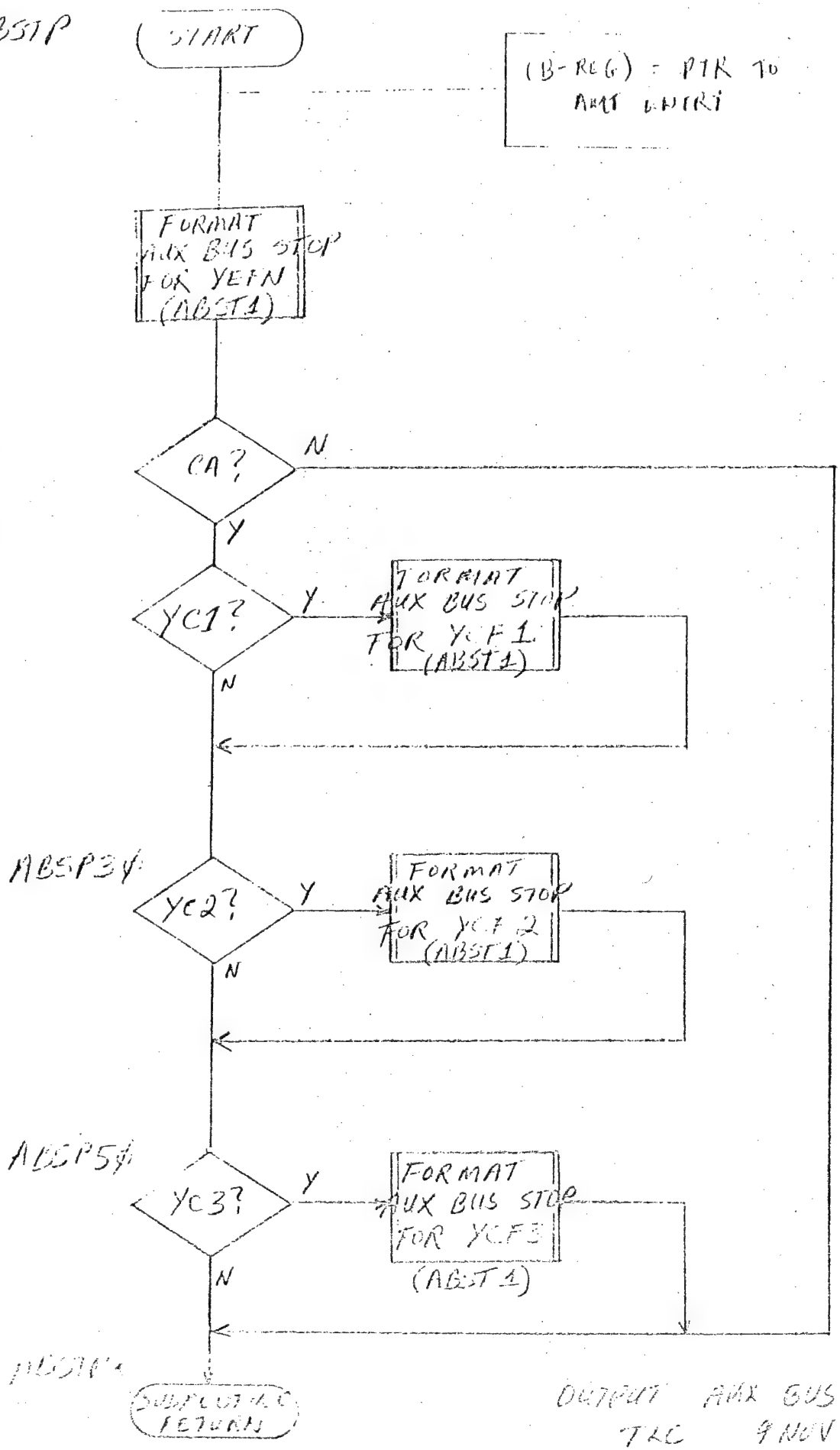
3.a.3.3.3.1



ABI MCT 2

RETURN DATA BLOCK(S)

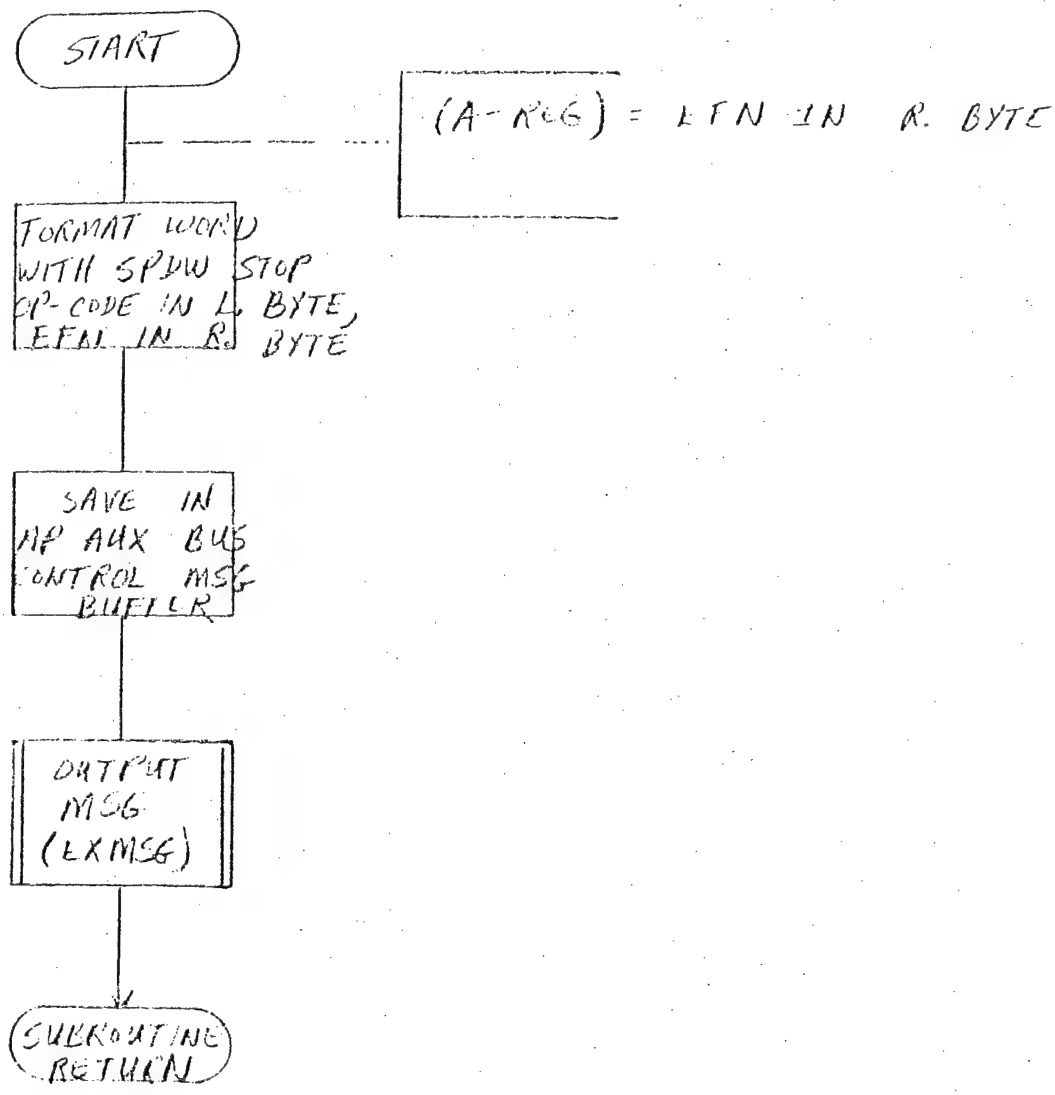
TLC 16 NOV 76



OUTPUT AUX BUS STOP MSGS
TLC 9 NOV 76

3.2.3.3.3.2

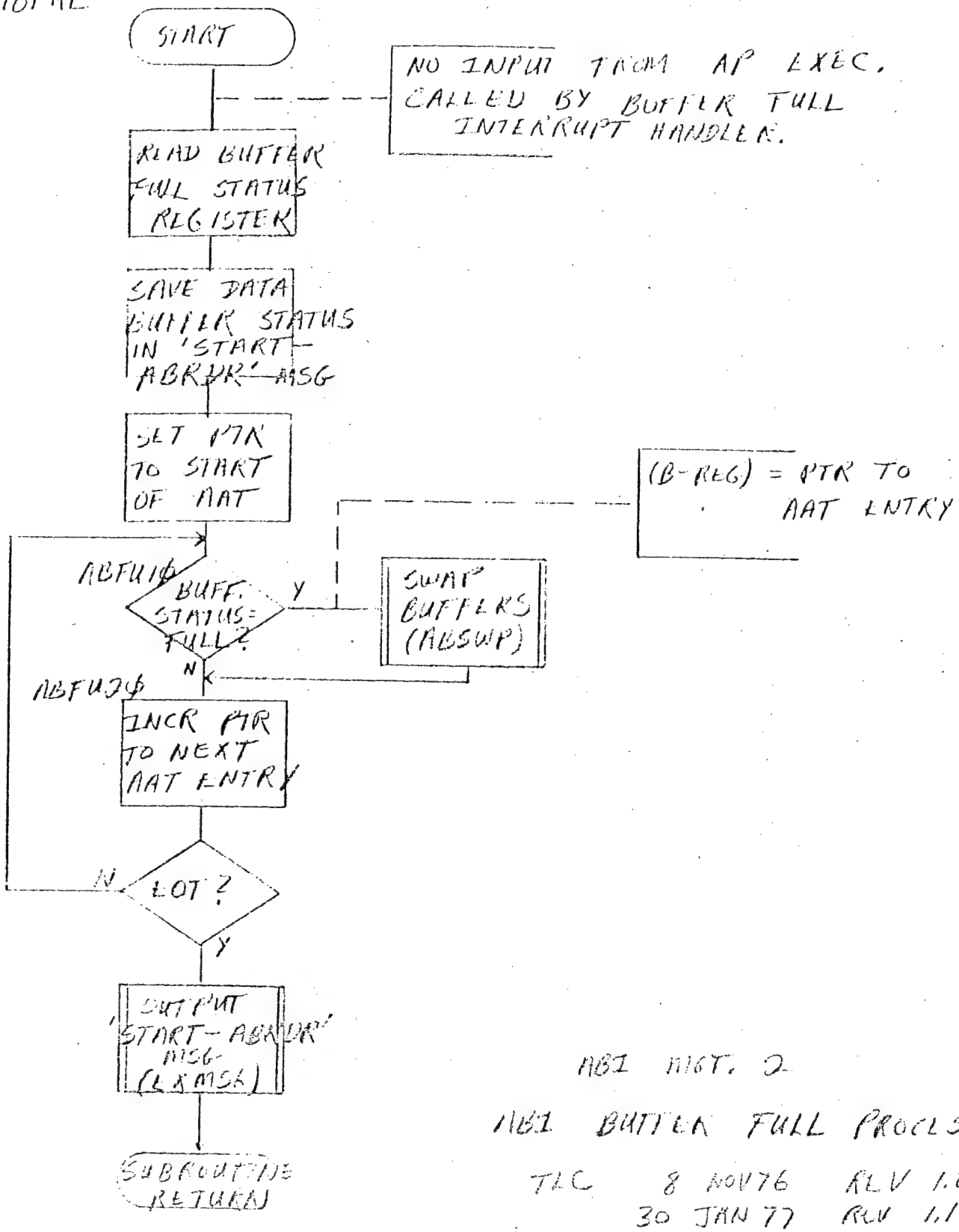
ABST 1



OUTPUT 1 AUX BUS STOP MESSAGE

TLC 30 JAN 77

ABTRL

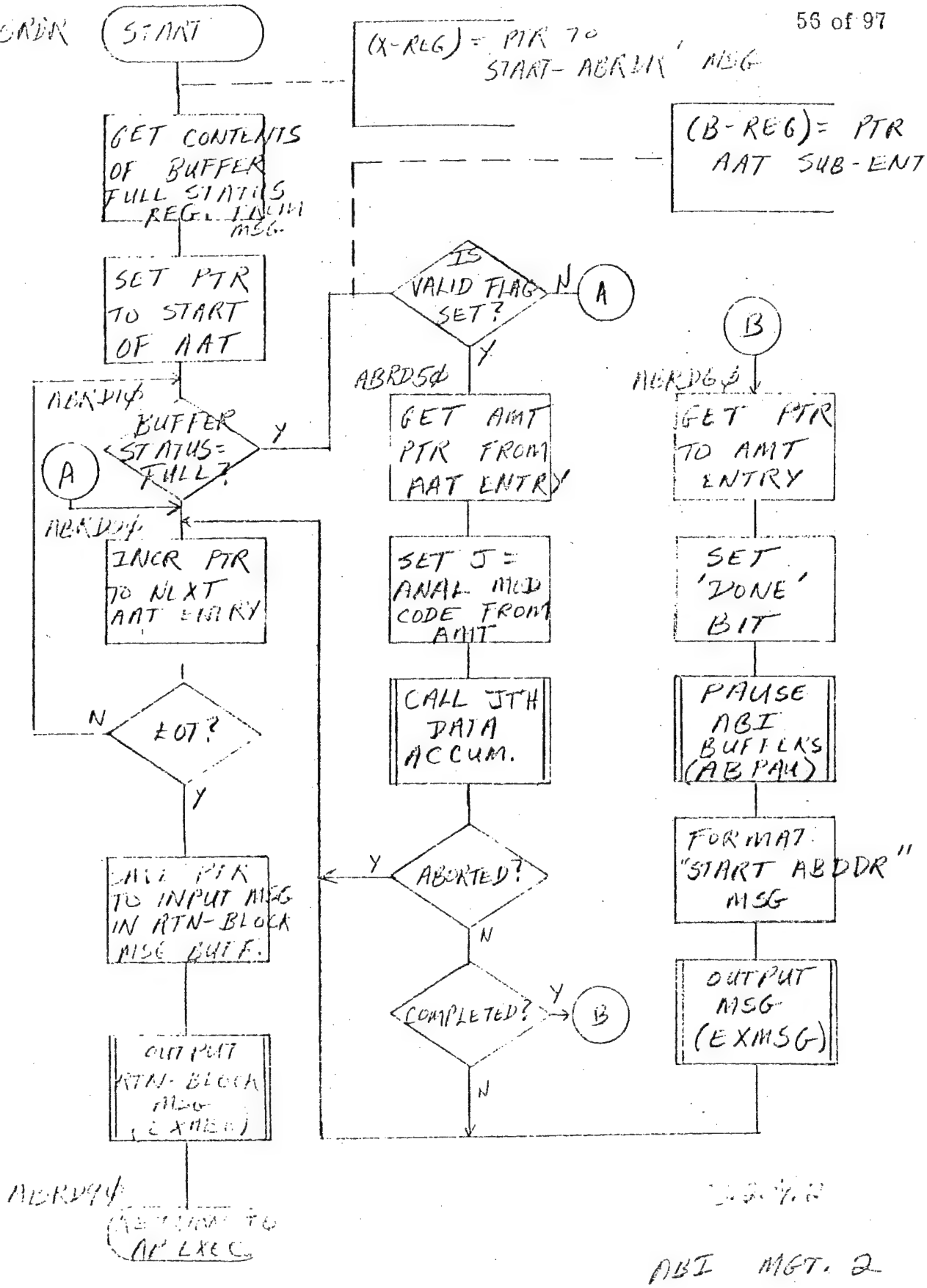


ABI MGT. 2

ABI BUFFER FULL PROCESSING

TLC 8 NOV 76 REV 1.0
30 JAN 77 REV 1.1

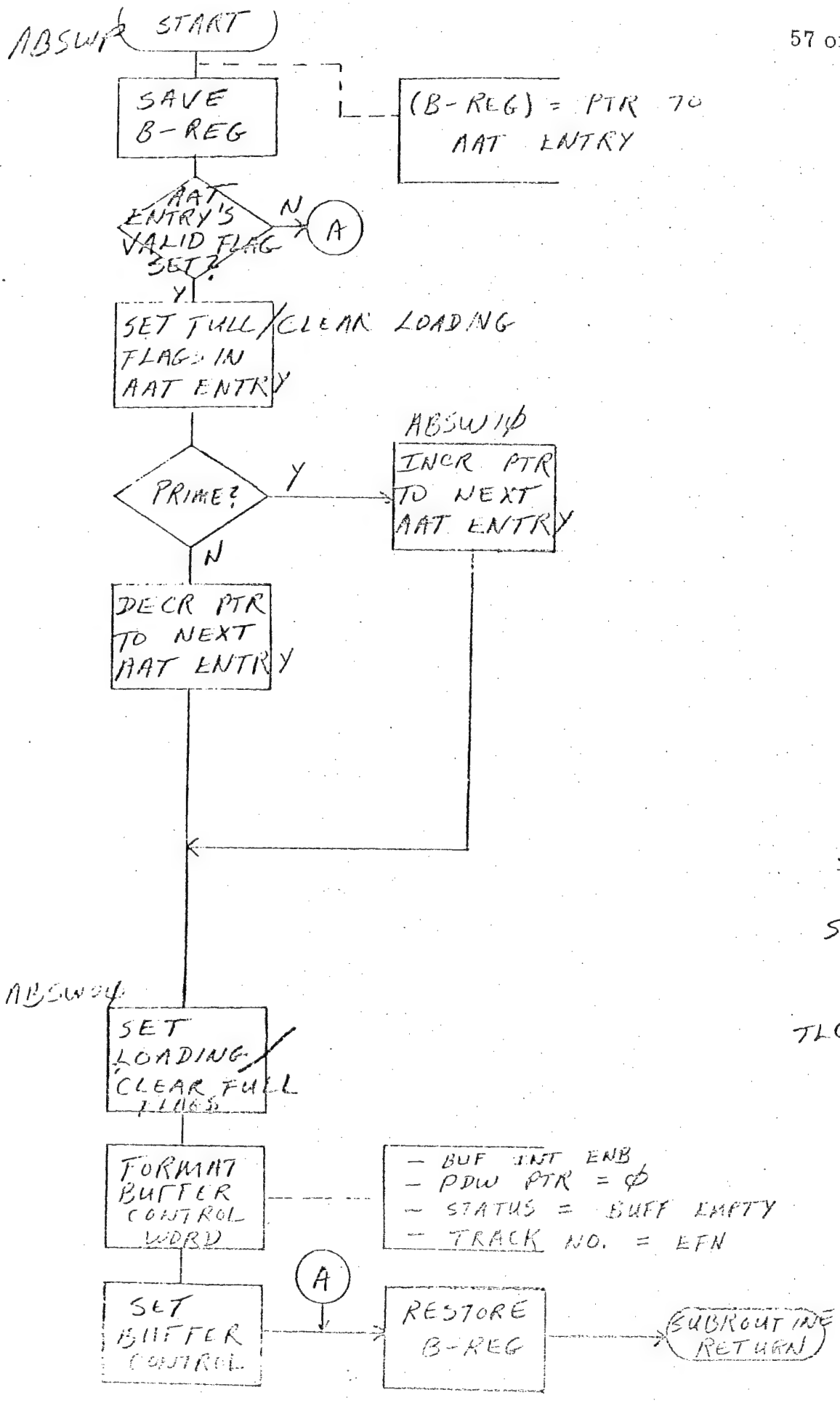
3.2.4.1



ABI MGT. 2

ABI RETURN DRIVER

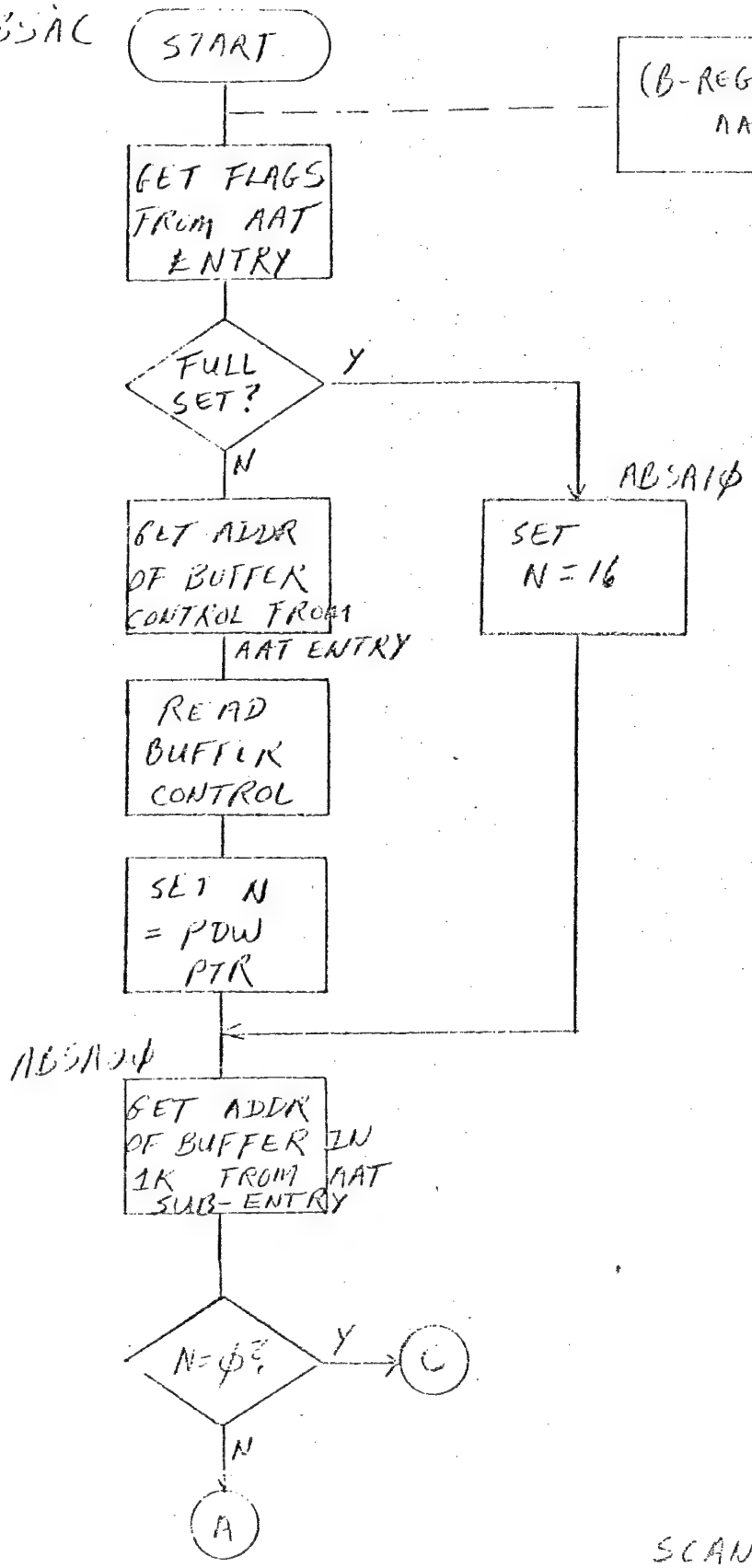
TLC 8 NOV 76 REL 1.0
31 JAN 77 REV 1.1



5.2.4.3

SWAP ABI BUFFERS

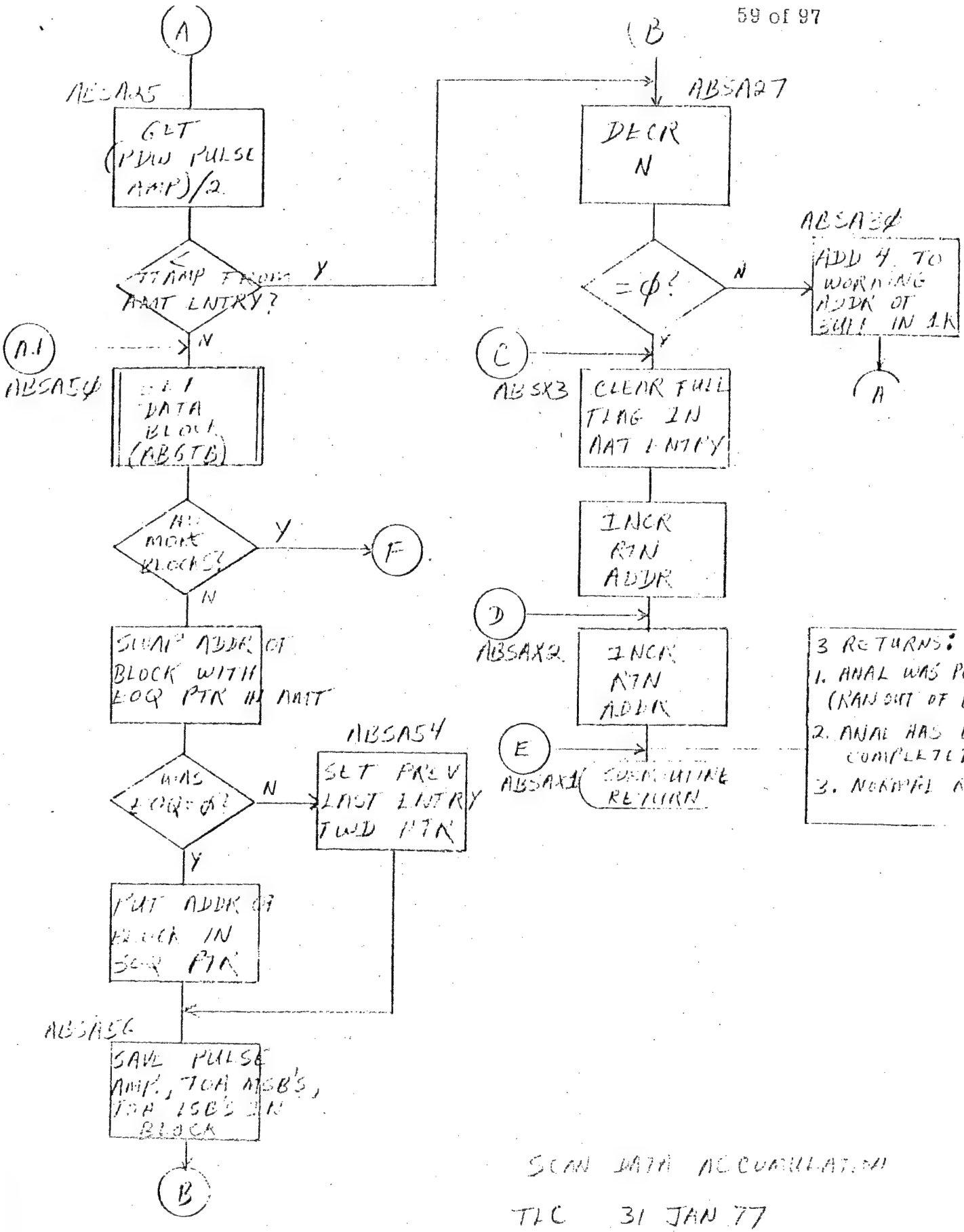
TLC 8NOV76



3.2.4.4

SCAN DATA ACCUMULATION

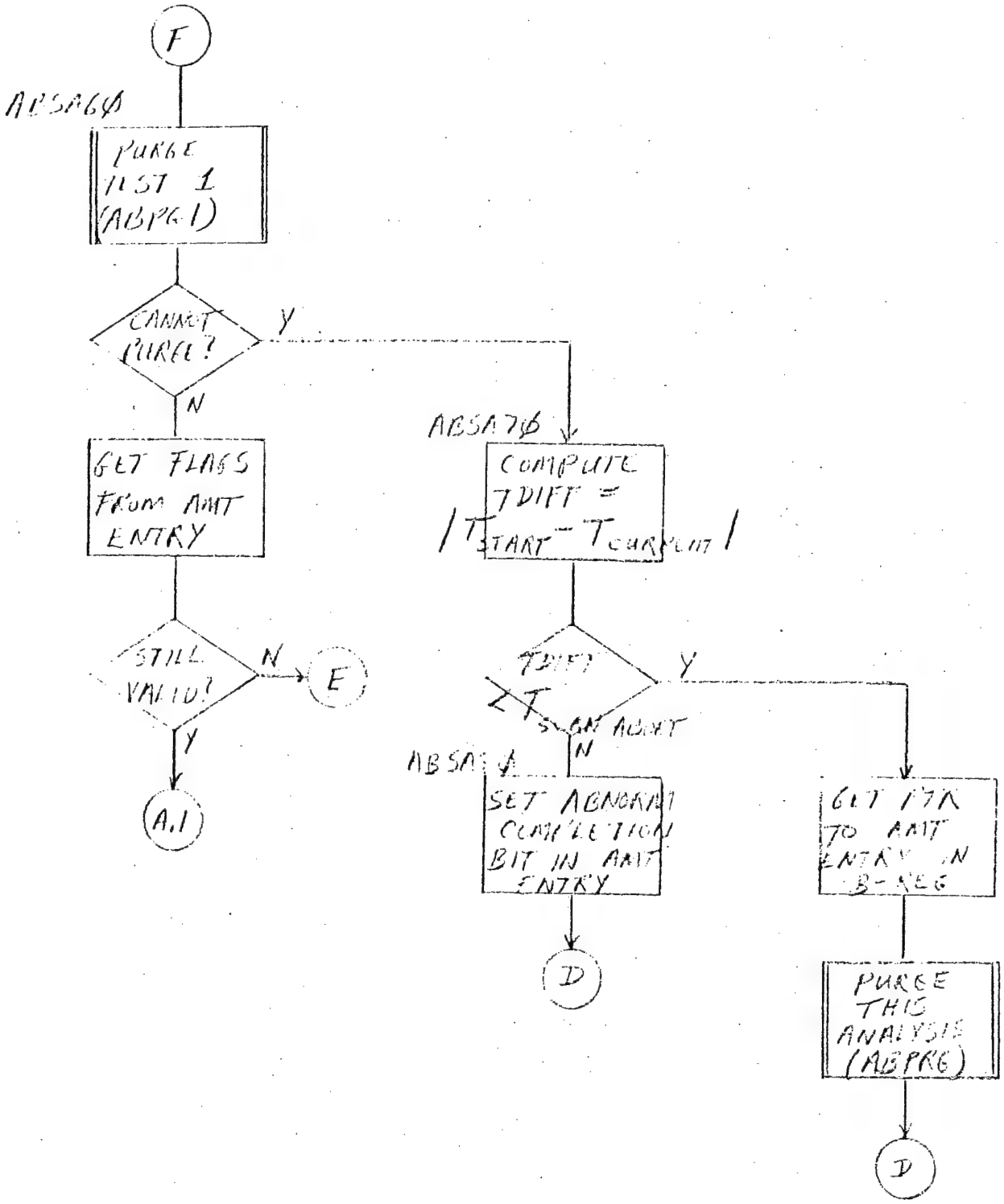
TLC 10 NOV 76 REV 1.0
21 JAN 77 REV 1.1



SCAN WITH ACCUMULATION

TLC 31 JAN 77

2 of 5

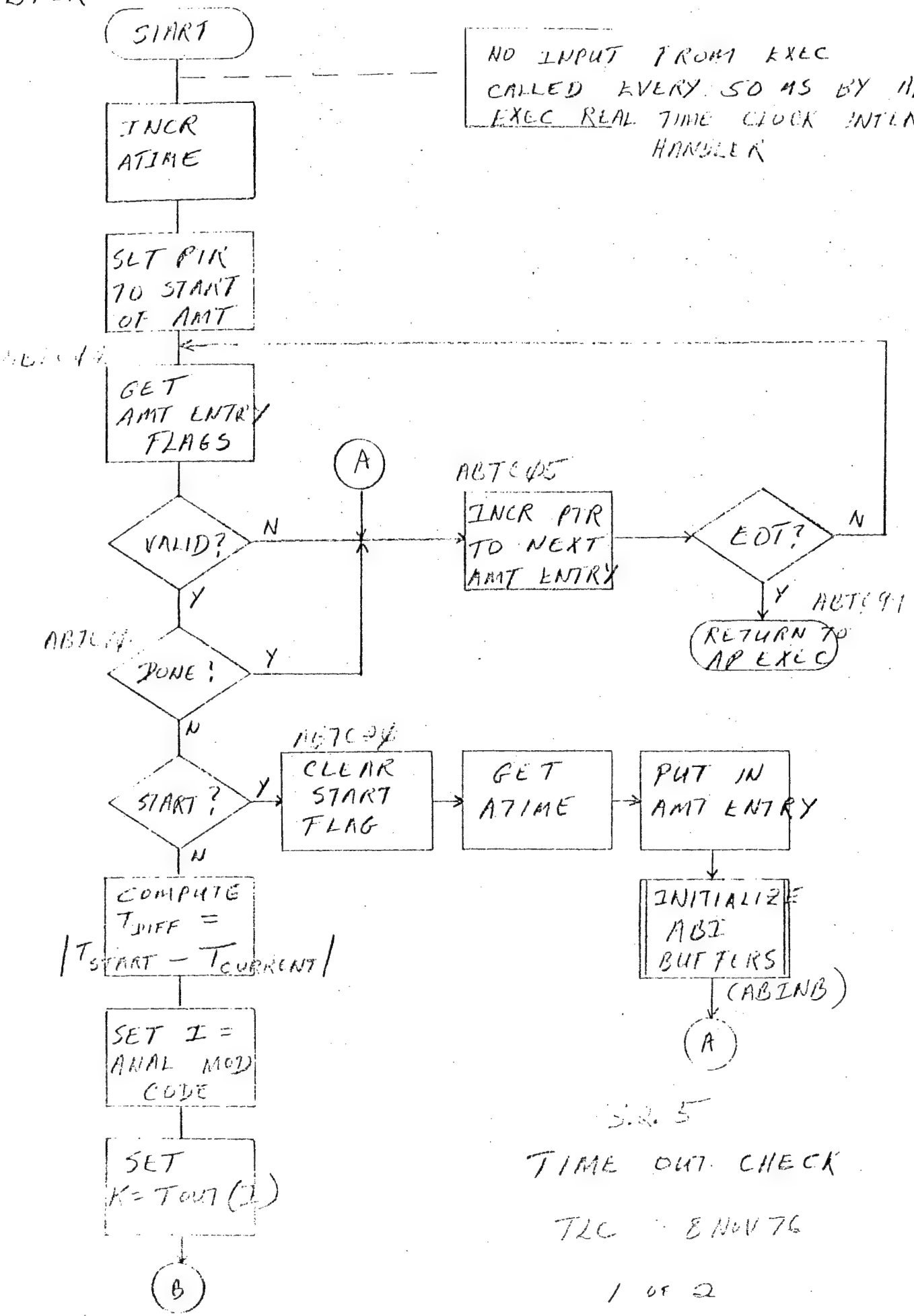


SCAN DATA ACCUMULATION

TLC 31JAN77

3 OF 3

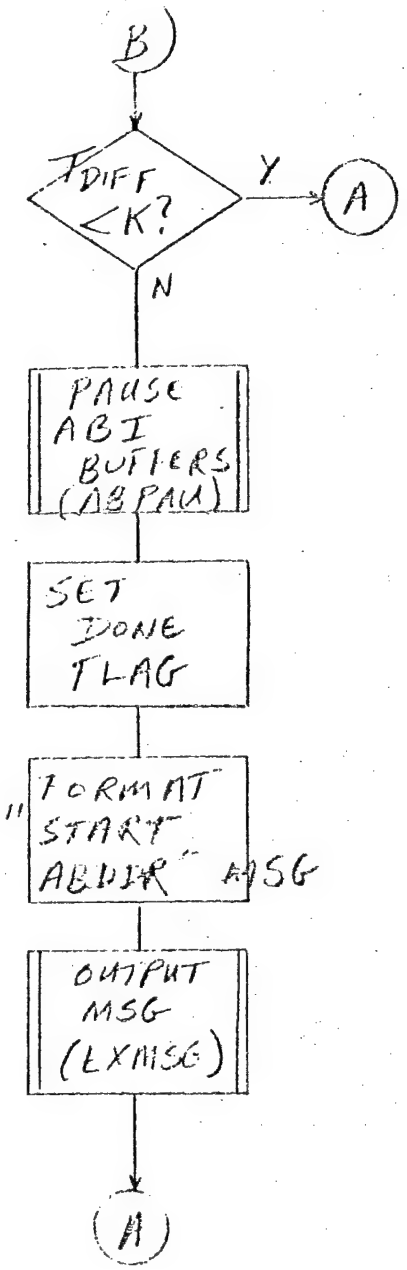
ABTCK



3.2.5

TIME OUT CHECK

TLC - 8 NOV 76

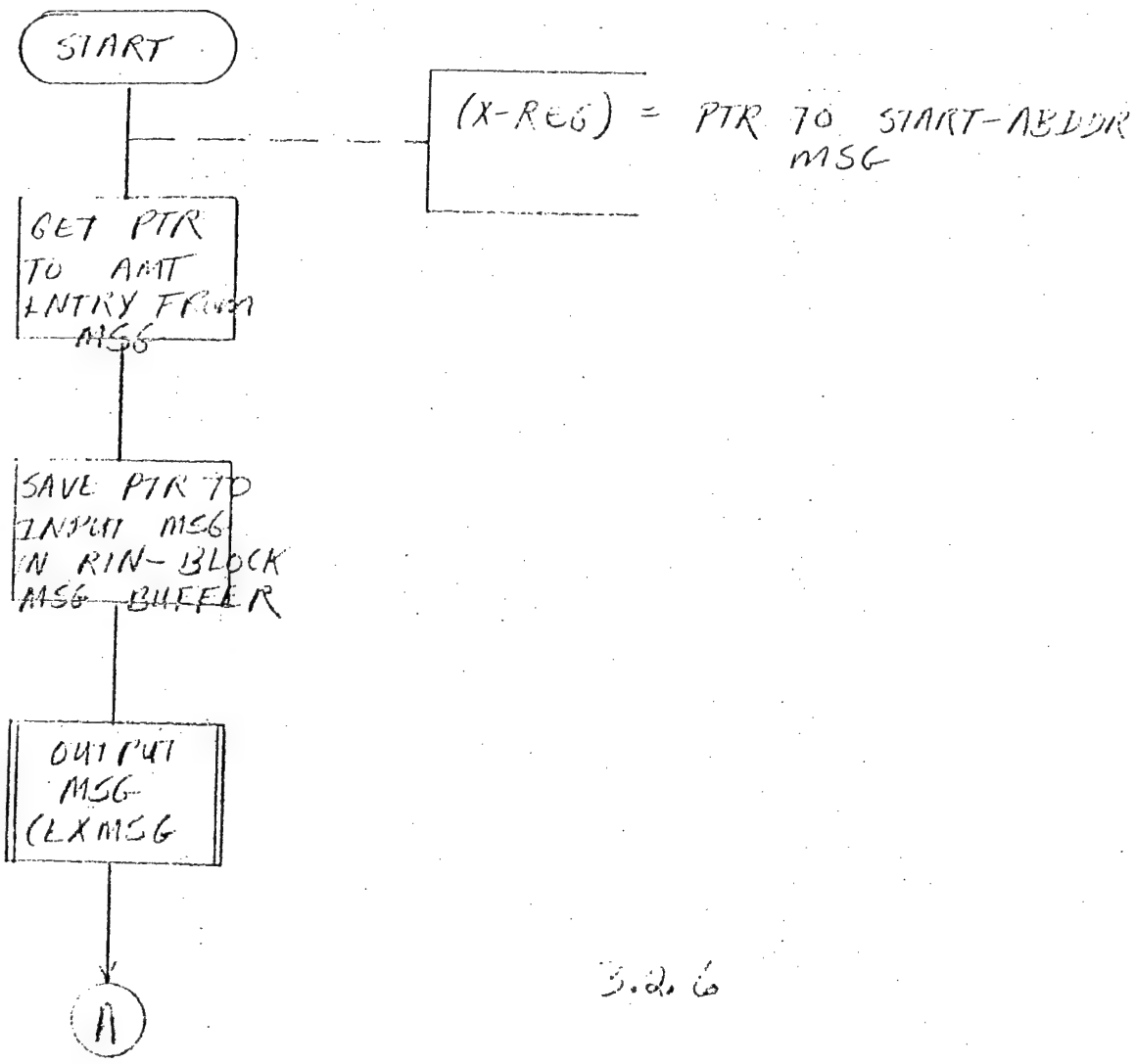


TIME OUT CHECK

TLC 8 NOV 76

2 OF 2

ABDDR



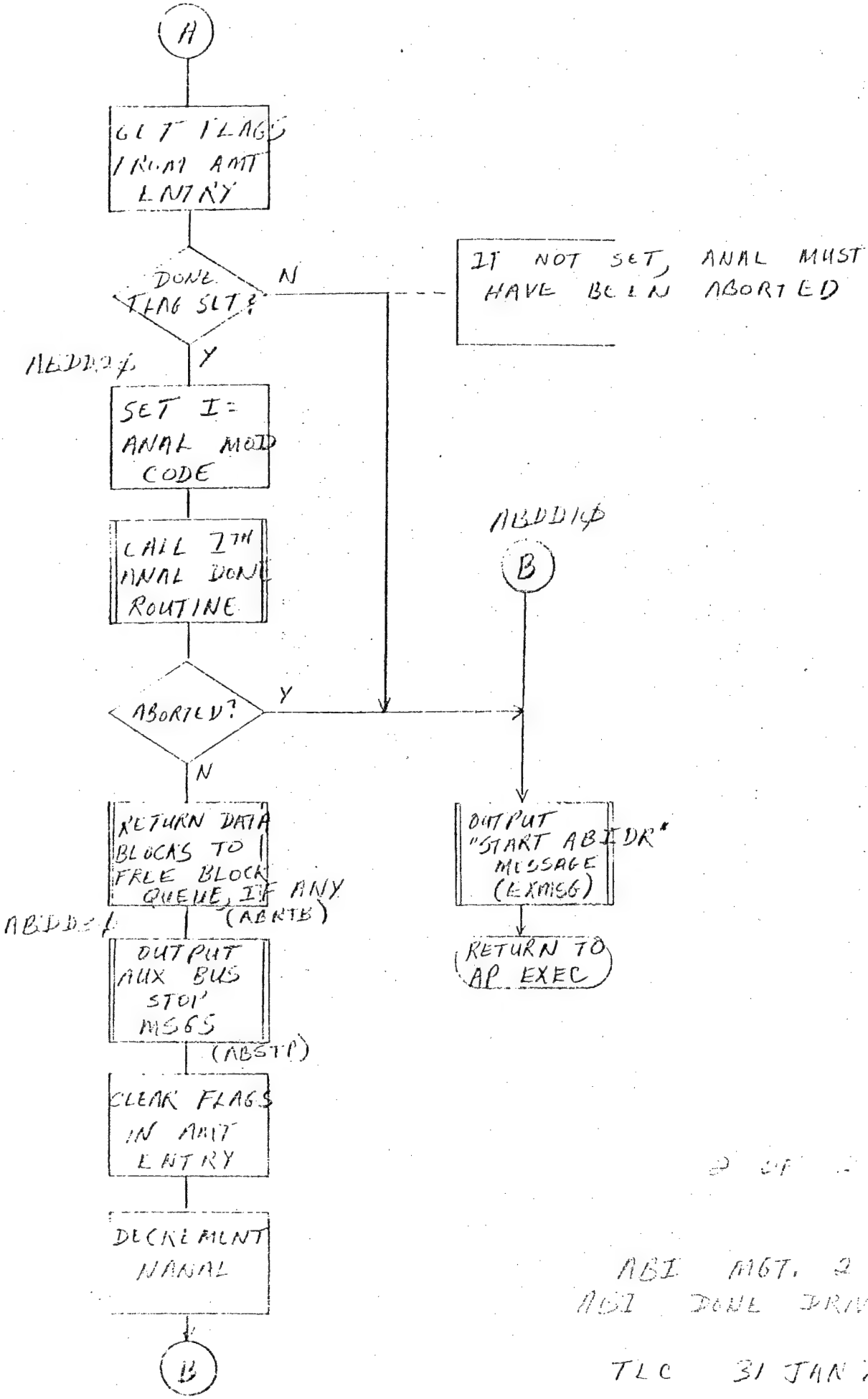
3.2.6

1 OF 2

ACT MGT. 2

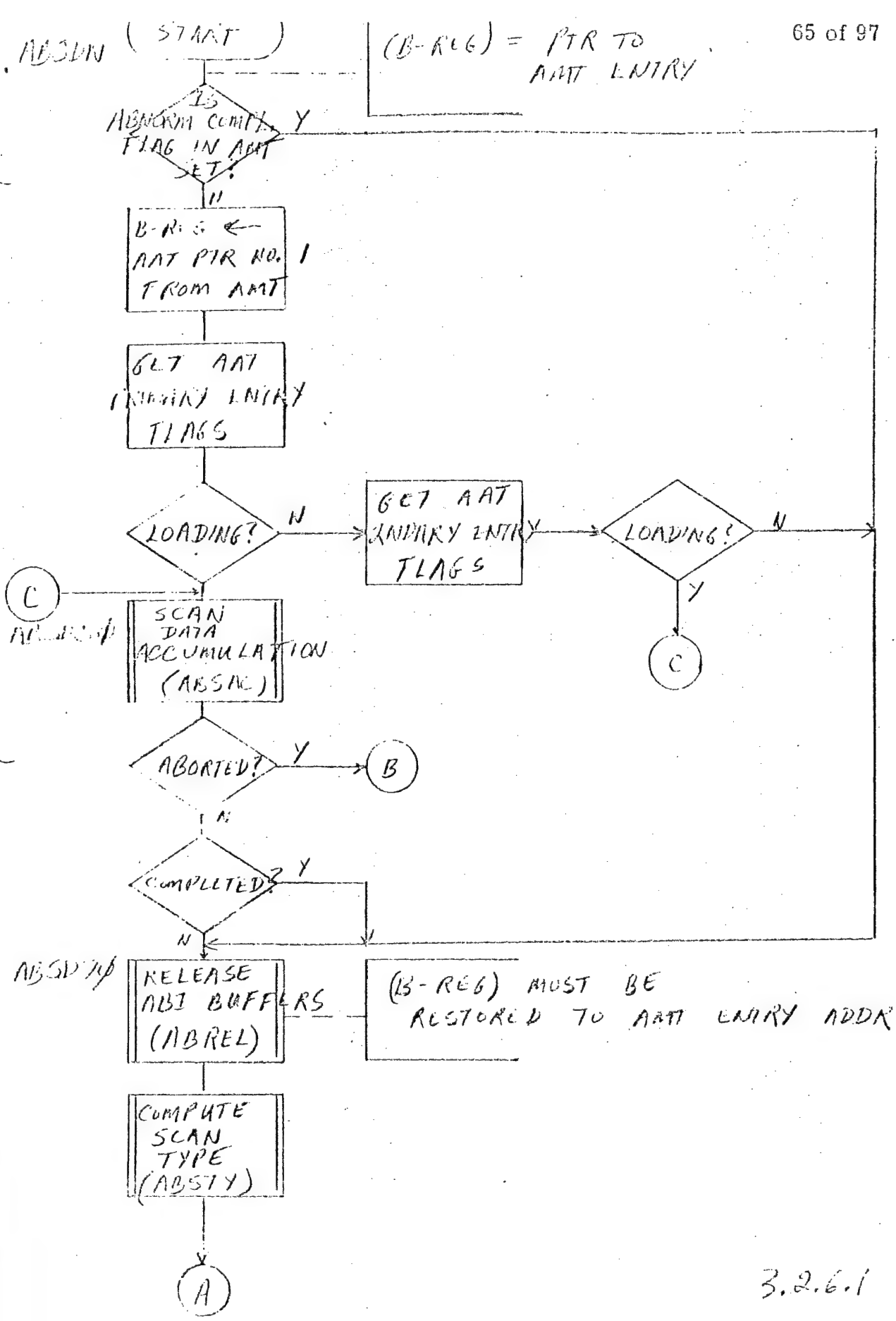
ACT DONE DRVER

TLC 10 NOV 76 REV 1.0
31 JAN 77 REV 1.1



ABI MGT. 2
ADI DONE DRIVER

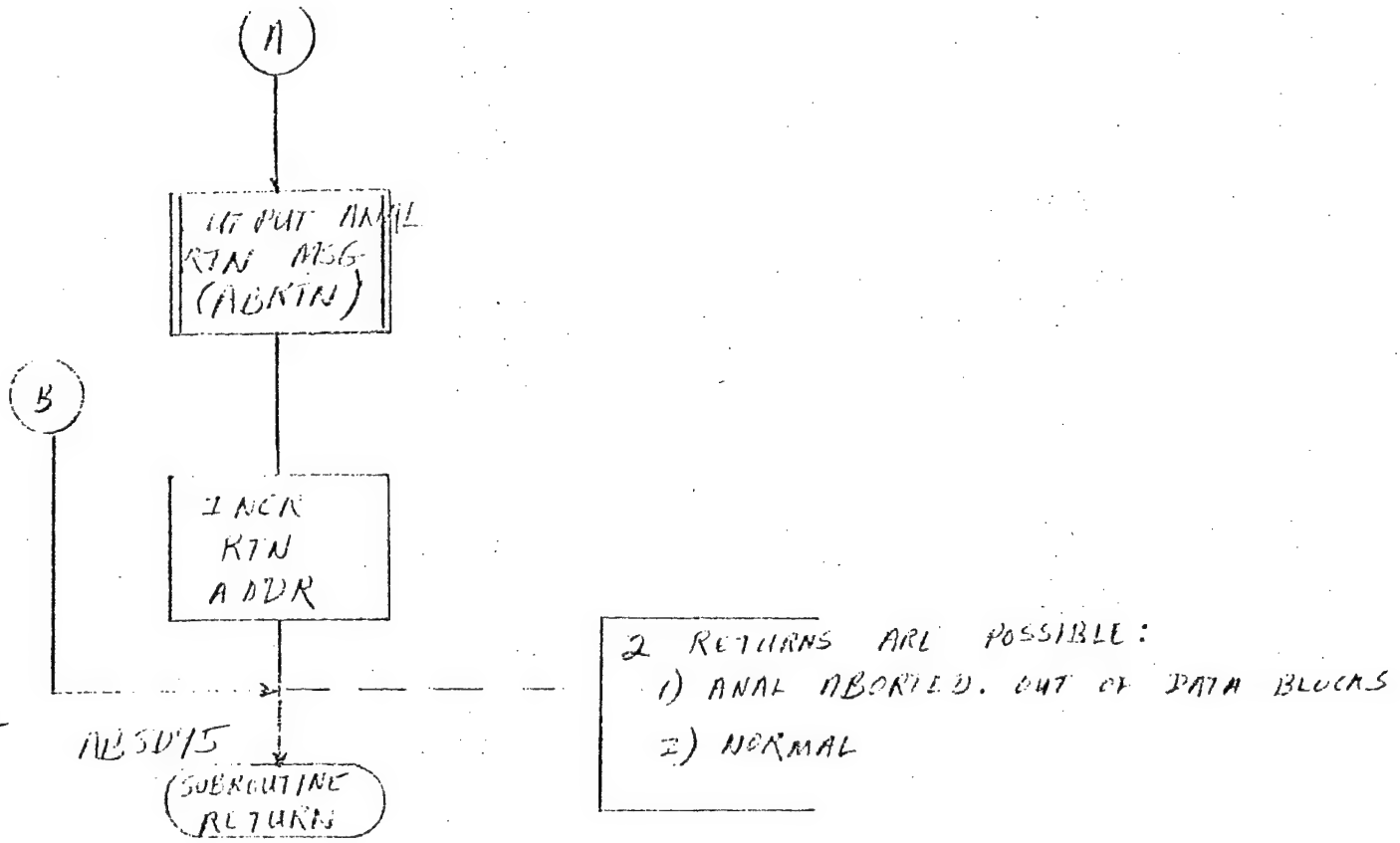
TLC 31 JAN 77



3.2.6.1

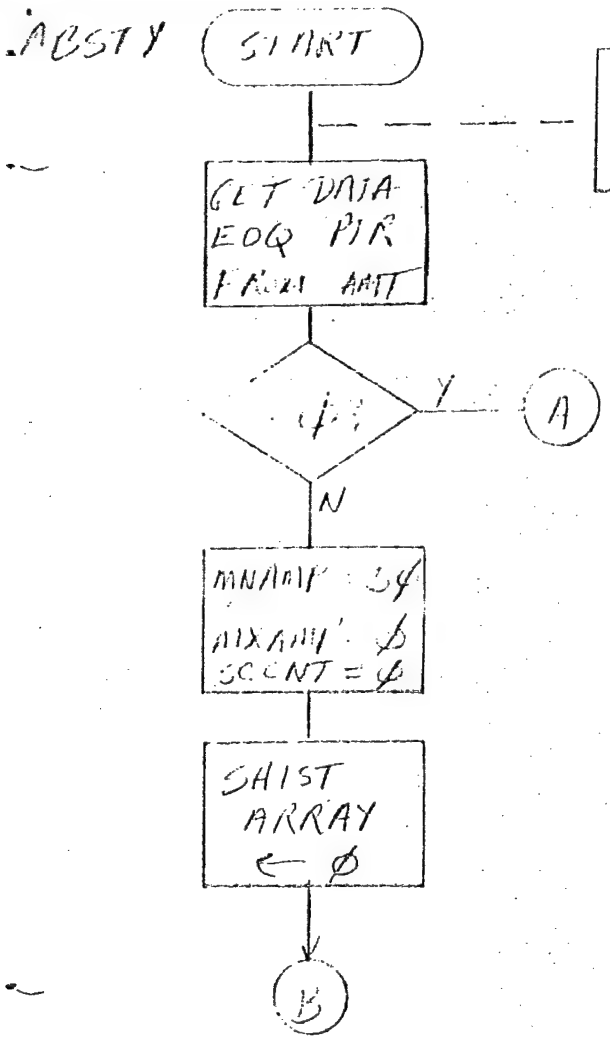
SCAN ANALYSIS DONE
TIC 7 NOV 76
31 JAN 77 REV

1 OF 2

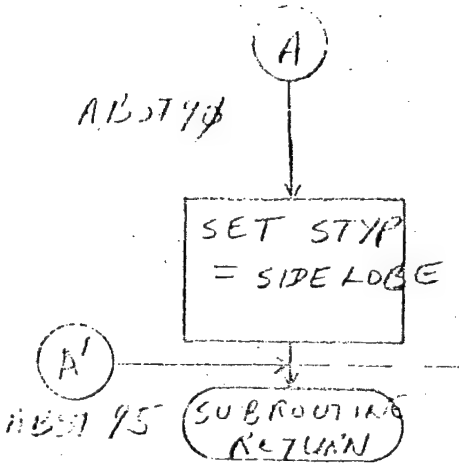


SCAN ANALYSIS DONE
TLC 31 JAN 77

2 OF 2



SHIST = AMP HISTOGRAM
MIXAMP = MAX AMP RECEIVED
MNAMP = MIN AMP RECEIVED
M2AMP = MAX ADJUSTED AMP REC'D
SCCNT = PDW COUNT
SCA1 = $\bar{A}$
SCA2 = $\overline{A^2}$
SCN = N
SCOM = σ^2_A
SCTS = t_s
SCK1 = k_1
SCK2 = k_2
STYP = SCAN TYPE



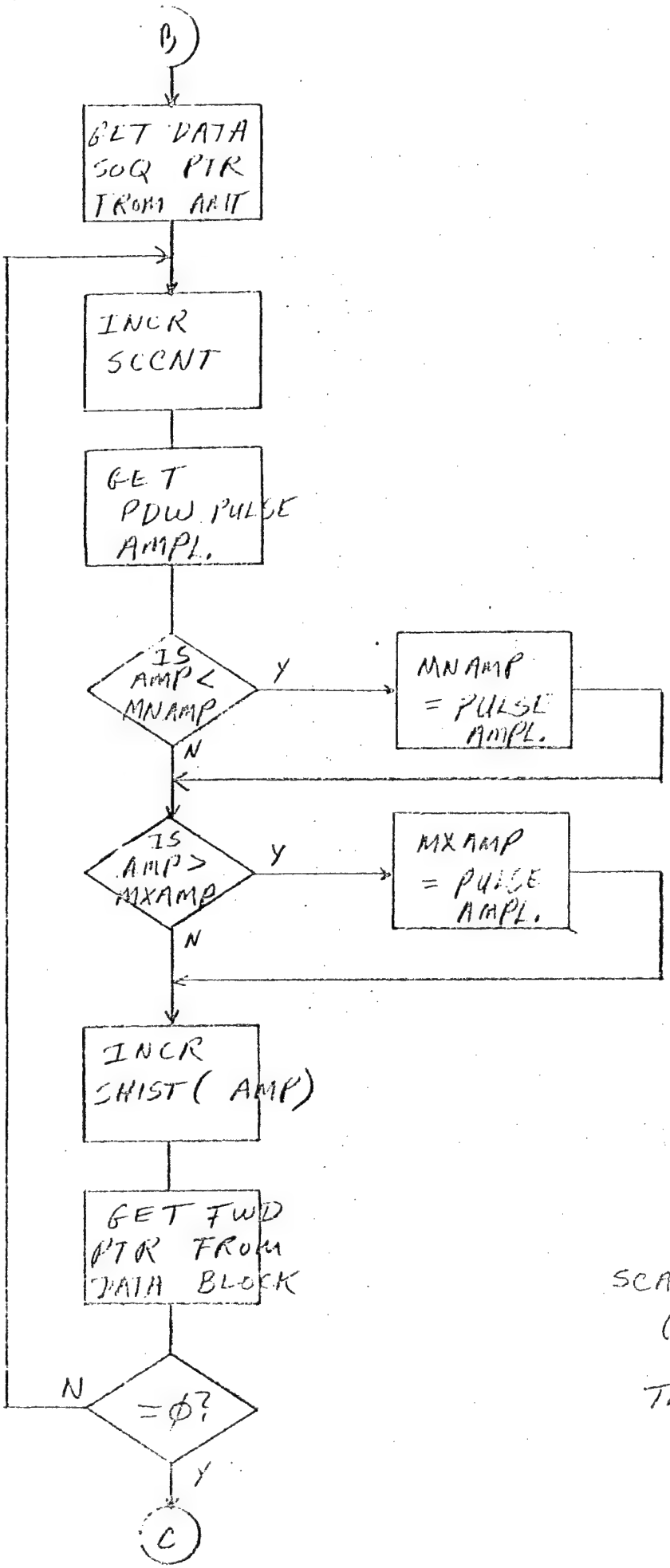
(A-REG) = STYP IN 4 MSB'S
0'S IN LSB'S

6.1.1

SCAN TYPE CALCULATION

TLC 17NOV 76

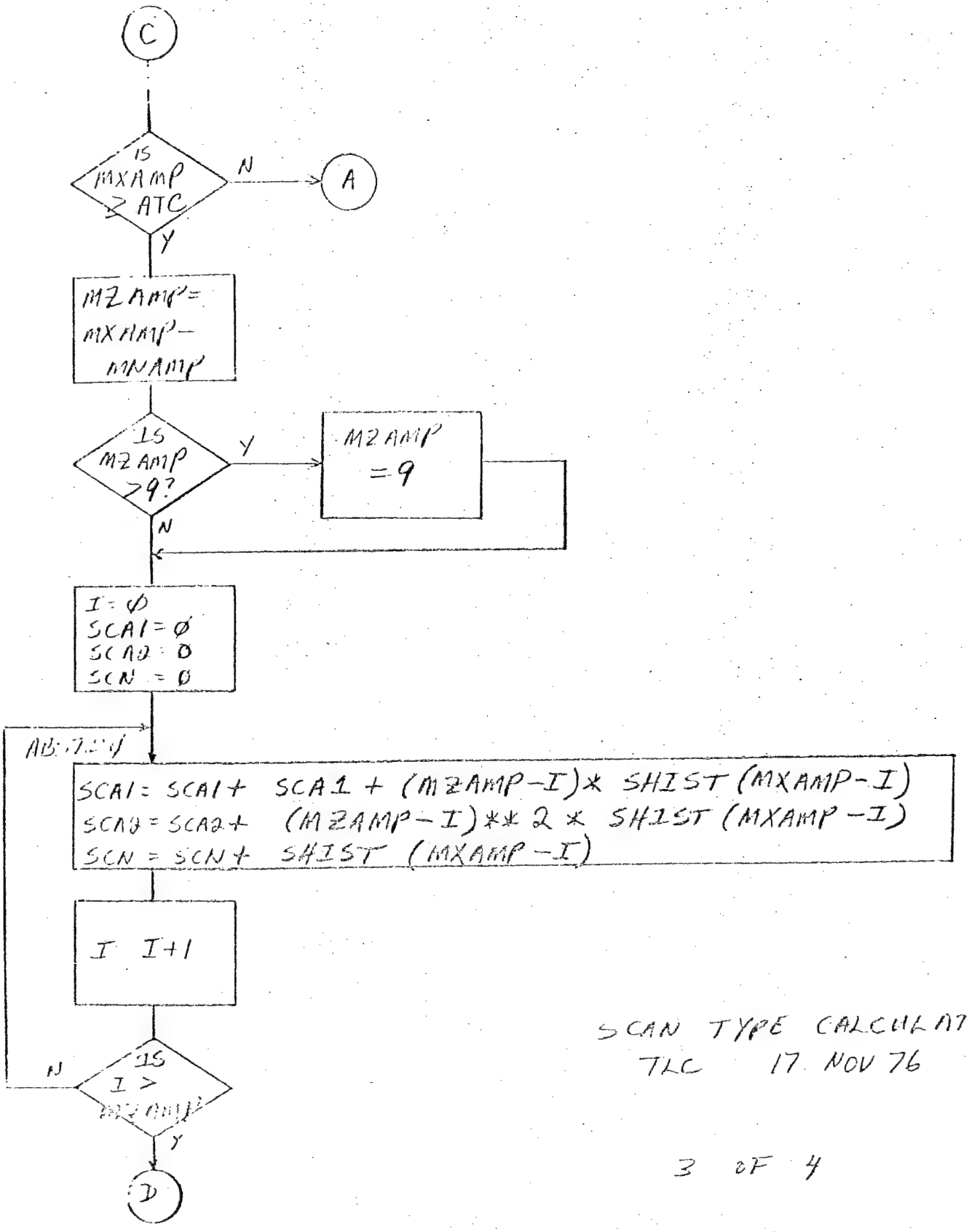
1 OF 4



SCAN TYPE CALCULATION
(HISTOGRAM FORMATION)

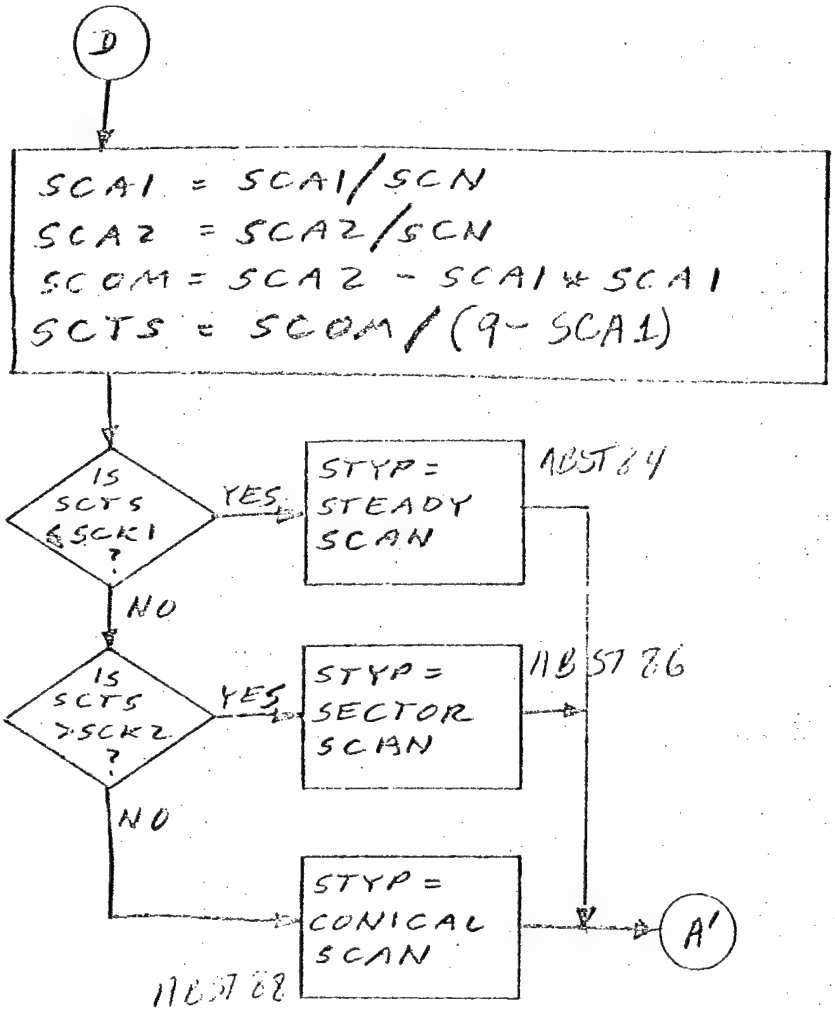
TLC 17 NOV 76

2 OF 4



SCAN TYPE CALCULATION
TLC 17 NOV 76

3 OF 4



SCAN TYPE CALCULATION

TLC 17NOV 76

4 OF 4

3.3 COMPUTER SUBPROGRAM ENVIRONMENT

3.3.1 Tables

3.3.1.1 Analysis Queue Pointers -

a) Table Name: ABQS and ABQE

b) Purpose and Type: Fixed length table containing the pointers to the first and last entries for each of the four queues.

c) Size and Indexing Procedure: Two sets (SOQ pointers and EOQ pointers) of four entries. Each entry shall be one 16-bit word. All SOQ pointers shall be referenced by indexed displacement from ABQS (Word Ø). All EOQ pointers shall be referenced by indexed displacement from ABQE (Word 4).

d) Entry Format: See Figure 1.

3.3.2 Variables

See Table 1.

3.3.3 Constants

See Table 2.

3.3.4 Flags

None.

3.3.5 Indices

None.

3.3.6 Common Data Base References

See Table 3.

ANALYSIS QUEUE POINTERS

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Word Ø	SOQ Ø															
1	SOQ 1															
2	SOQ 2															
3	SOQ 3															
4	EOQ Ø															
5	EOQ 1															
6	EOQ 2															
7	EOQ 3															
8	NOT APPLICABLE															
9																
10																
11																
12																
13																
14																
15																

Figure 1a Table Format

Field	Description	Units	LSB
SOQ Ø	Priority Ø Analysis SOQ Pointer	N/A	N/A
EOQ Ø	" Ø " EOQ "	↓	↓
SOQ 1	" 1 " SOQ "		
EOQ 1	" 1 " EOQ "		
SOQ 2	" 2 " SOQ "		
EOQ 2	" 2 " EOQ "		
SOQ 3	" 3 " SOQ "		
EOQ 3	" 3 " EOQ "		

Figure 1b.

TABLE 1

ABI MANAGEMENT 2 VARIABLES

Descript. Item	Name			
	SDQ	EDQ	ATIME	NBUF
Purpose	Start of queue pointer for free data blocks queue	End of queue pointer for free data blocks queue	ABI Management 2 Time Units are m sec LSB = 50 m sec	No. of ABI 1K RAM double buffers in use
Type	Pointer	Pointer	Fixed point	Fixed point
Size	16 Bits	16 Bits	16 Bits	16 Bits
Binary Pt.	N/A	N/A	Bit 0	Bit 0
Max. Value	N/A	N/A	65536	MAX BUF
Min. Value	N/A	N/A	0	ø
Initial Value	SDBLK	EDBLK	0	ø

TABLE 1
-continued - ABI MANAGEMENT 2 VARIABLES

Descript. Item	Name			
	NANAL			
Purpose	No. .of analyses in progress			
Type	Fixed point			
Size	16 Bits			
Binary Pt.	Bit 0			
Max. Value	MAXANL			
Min Value	0			
Initial Value	0			

TABLE 2

ABI MANAGEMENT 2 CONSTANTS

Descript. Item	Name			
	MAXBUF	MAXANL	MINAMP	
Purpose	No. of ABI 1K RAM double buffers in IEWS, ADM	Maximum no. of analyses in progress	Minimum value of TTAMP for a To-Sorter SPDW Request Units are DBM LSB = 3.2	
Type	Fixed point	Fixed point	Fixed point	
Size	16 Bits	16 Bits	16 Bits	
Binary Pt.	Bit 0	Bit 0	Bit 0	
Initial Value	8	8	2	

TABLE 2

-continued - ABI MANAGEMENT 2 CONSTANTS

Descript. Item	Name			
	SDBLK	EDBLK		
Purpose	Starting address of AP memory assigned to data blocks (Low memory address)	End address of AP memory assigned to data blocks (High memory address)		
Type	Memory Address	Memory Address		
Size	16 Bits	16 Bits		
Binary Pt.	N/A	N/A		
Initial Value	Computed by assembler	Computed by assembler		

TABLE 3

COMMON DATA BASE REFERENCES

Common Data Base Item	Major Routine (Including Supporting Routines)					
	AB1DR	AB2DR	ABIDR	ABRDR	ABTCK	ABDDR
ETF	U					
ATC						U
AUXMT	B		U			
AMT			B	B	B	B
AAT			B	B		B

S = Set

U = Used

B = Both

3.3.7 Queues

3.3.7.1 Data Block Queues -

Data shall be accumulated for all analyses in progress by attaching FIFO queues to the AMT entry for the analysis. The queue entries shall be 4-word blocks. A queue of unused (free) blocks shall be created by the AB2IN initialization routine. The blocks attached to the AMT entry queues shall be obtained from this free block queue. The format of the queue entry for the queues attached to AMT entries is shown in Figure 2.

The format of free block queue entries shall be identical except that only the FWD PTR field is valid. The structure of these queues is shown in Figure 3.

3.3.7.2 Analysis Queues -

There shall be four analysis queues used to buffer incoming analysis start messages. Each queue shall correspond to a analysis priority level. The queue entries shall be 22-word blocks and shall be executive message blocks. Drivers in the ABI1 and ABI2 Functional Groups shall assume the responsibility of returning message blocks to the EXEC's free message block queue. Analysis queue entry format is shown in Figure 4.

Word	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Ø	FWD PTR															
1	PAMP					X			TOA MSBS				X			
2	TOA LSBS															
3	NOT USED															
4	NOT APPLICABLE															
5																
6																
7																
8																
9																
10																
11																
12																
13																
14																
15																

Figure 2a. Data Queue Entry Format

Field	Description	Units	LSB
FWD PTR	Pointer to next block in queue. If last block, FWD PTR = Ø	N/A	N/A
PAMP	Pulse amplitude from PDW	DBM	1.6
TOA MSBS	Pulse Time of Arrival (MSB's)	μsec	2^{16}
TOA LSBS	Pulse Time of Arrival (LSB's)	μsec	1

Figure 2b.

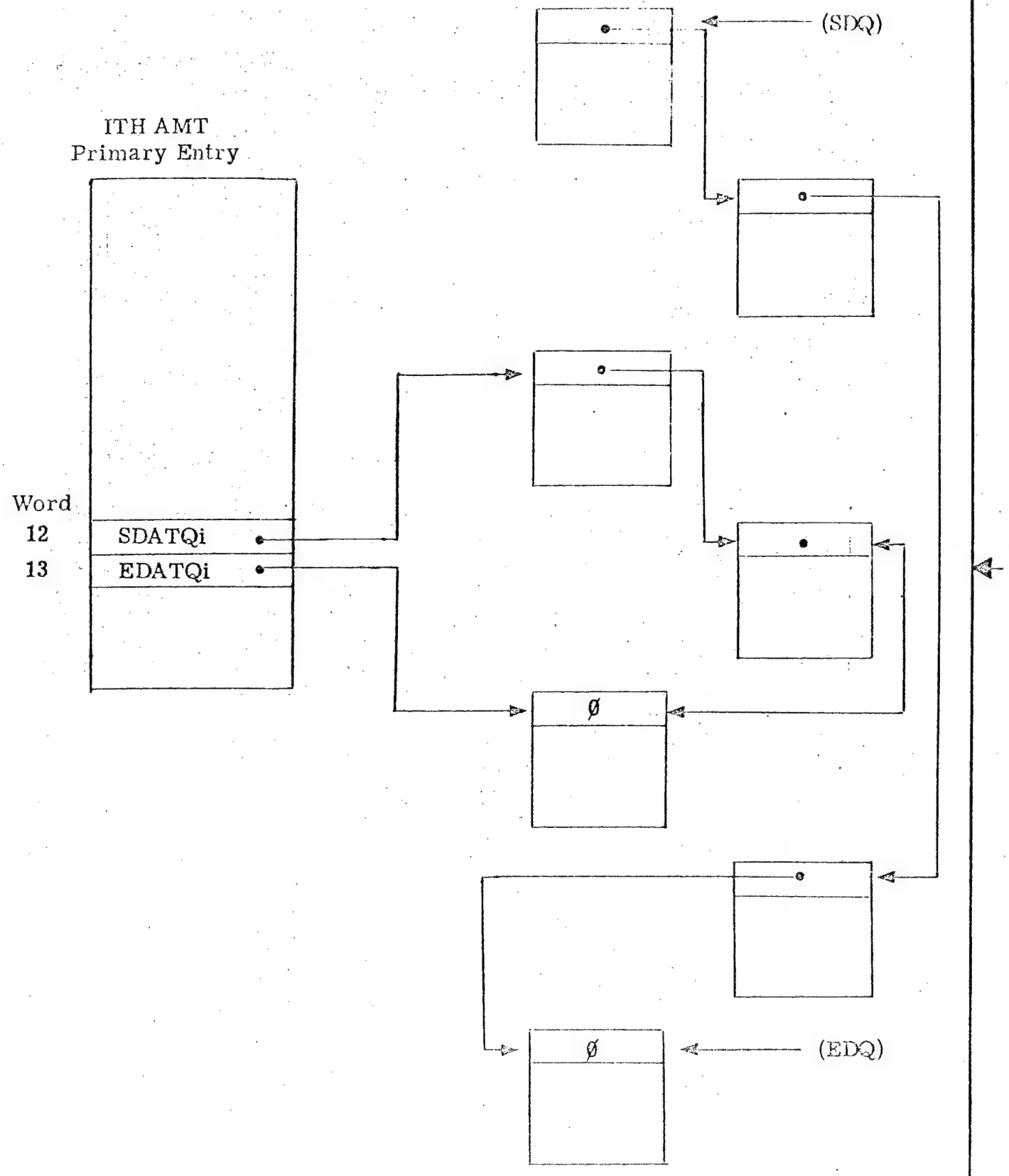


Figure 3. Example of Free Data Block Queue & ITH AMT Entry Data Queue

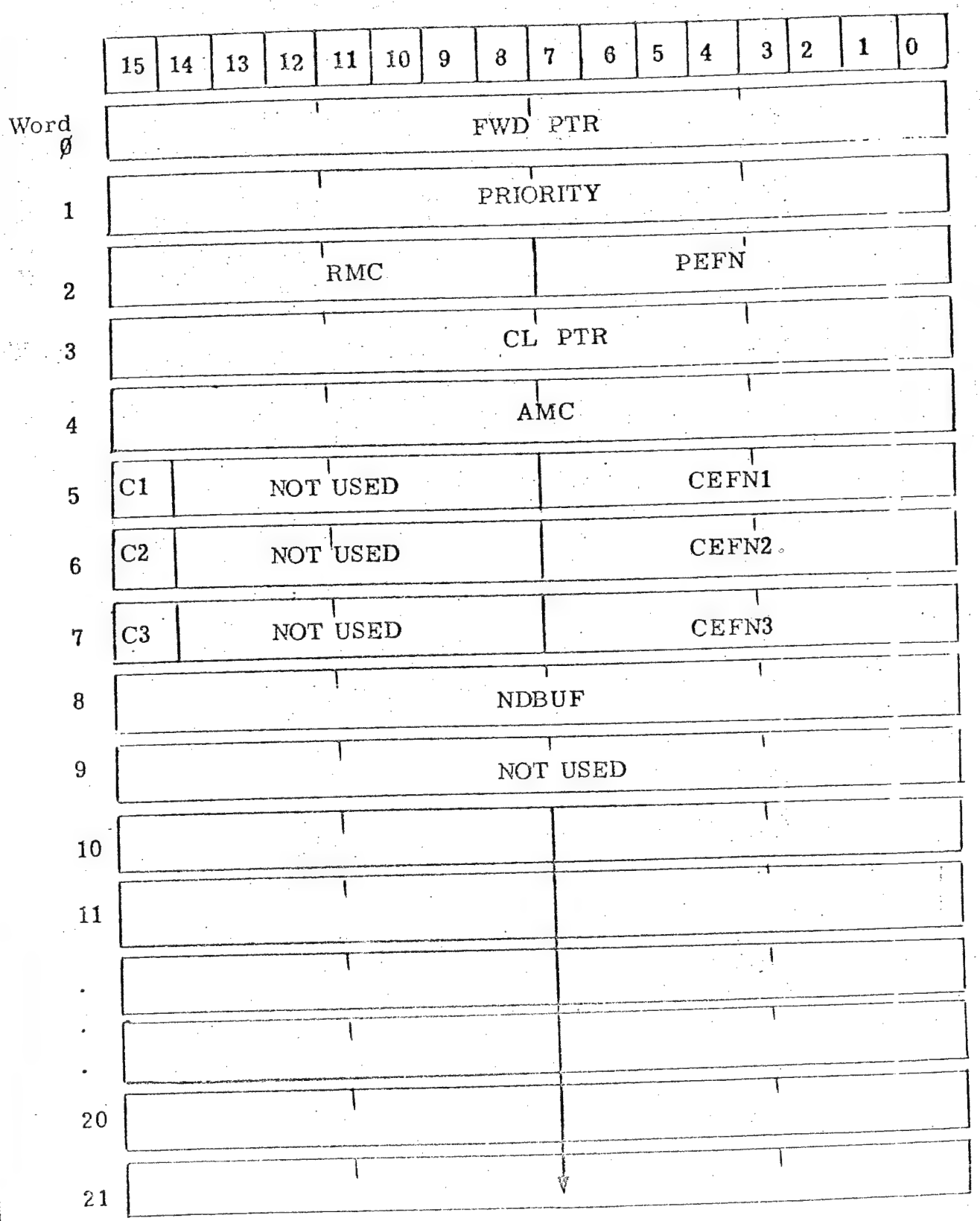

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
Word Ø
FWD PTR
1
PRIORITY
2
RMC
PEFN
3
CL PTR
4
AMC
5
C1
NOT USED
CEFN1
6
C2
NOT USED
CEFN2
7
C3
NOT USED
CEFN3
8
NDBUF
9
NOT USED
10
11
20
21

Figure 4a.

Field	Description	Units	LSB
FWD PTR	Pointer to next block in Analysis Queue	N/A	N/A
PRIORITY	Priority assigned to analysis (= 0, 1, 2, or 3)	N/A	1
RMC	Return Module Code 1 NE Proc 2 2 NE Proc 3 3 Nofa 2 Proc 2 4 Nofa 2 Proc 3 5 EOC Proc 2 6 EOC Proc 3 7 EOC Proc 4 8 EC2 9 EC3		
PEFN	Primary EFN under analysis $0 \leq PEFN \leq 127$	N/A	1
CLPTR	Pointer to data base candidate list	N/A	N/A
AMC	Analysis Module Code 0 Scan 1 Frequency 2 PRI 3 Contemporaneous 4 Deinterleaving	N/A	N/A
C1	CEFN1 flag (CEPN1 is valid if C1=1)	N/A	N/A
CEFN1	Suspected contemporaneous EFN no. 1 $0 \leq CEFN1 \leq 127$	N/A	1
C2	CEFN2 flag	N/A	N/A
CEFN2	Suspected contemporaneous EFN no. 2	N/A	1
C3	CEFN3 flag	N/A	N/A
CEFN3	Suspected contemporaneous EFN no. 3	N/A	1
NDBUF	No. of double buffers required (in ABI 1K RAM: = 1, 2, 3, or 4)	N/A	1

Figure 4b.

3.4 INPUT/OUTPUT FORMATS

The format of instrumentation data output shall be as specified in the Data Extraction CSDD, 53959-GT-0759. The format of input and output Executive messages shall be as specified in the Common Data Base Design Document, 53959-GT-0751. The following message types shall be used:

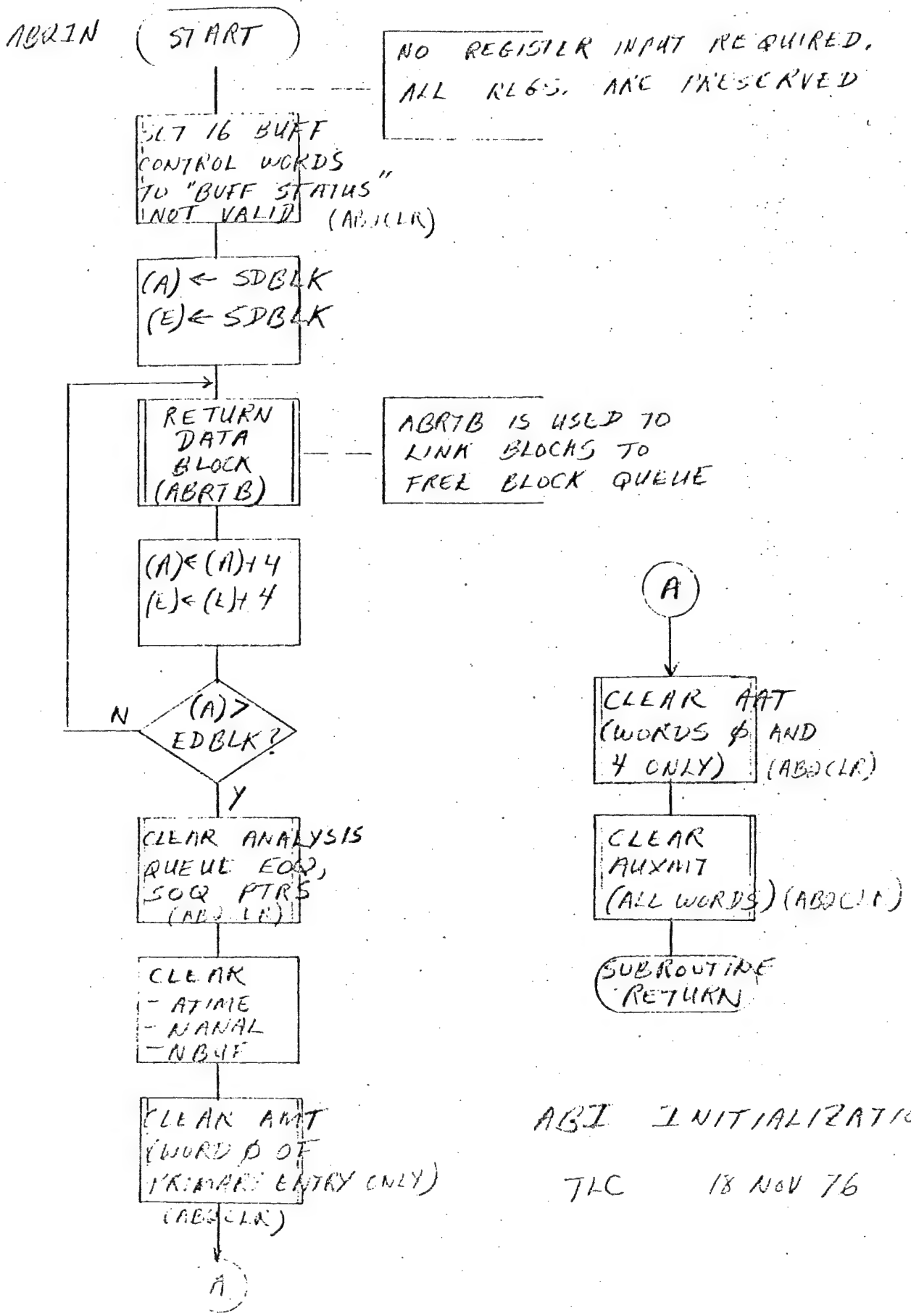
Executive Message No.	Message Name	Input or Output
1	Analysis Request	Input
2	Analysis Start	Both
3	RMP Aux Bus Control	Input
4	Analysis Return	Output
17	Start ABDDR	Both
19	Sorter Control	Output
21	AP Aux Bus Control	Both
22	Start ABIDR	Both

3.5 SYSTEM LIBRARY SUBROUTINES

There shall be no system library subroutines required by the ABI Management 1 and 2 Functional Groups.

3.6 CONDITIONS FOR INITIALIZATION

The ABI Management 1 Functional Group shall require that the SC and AGTG flags in each entry of the AUXMT be set to ∅. The ABI Management 2 Functional Group shall require that the ABI Management 2 Initialization routine (AB2IN) be executed to initialize queues, tables, etc., used by this functional group. The flow chart for AB2IN follows.



ABI INITIALIZATION

TLC 18 NOV 76

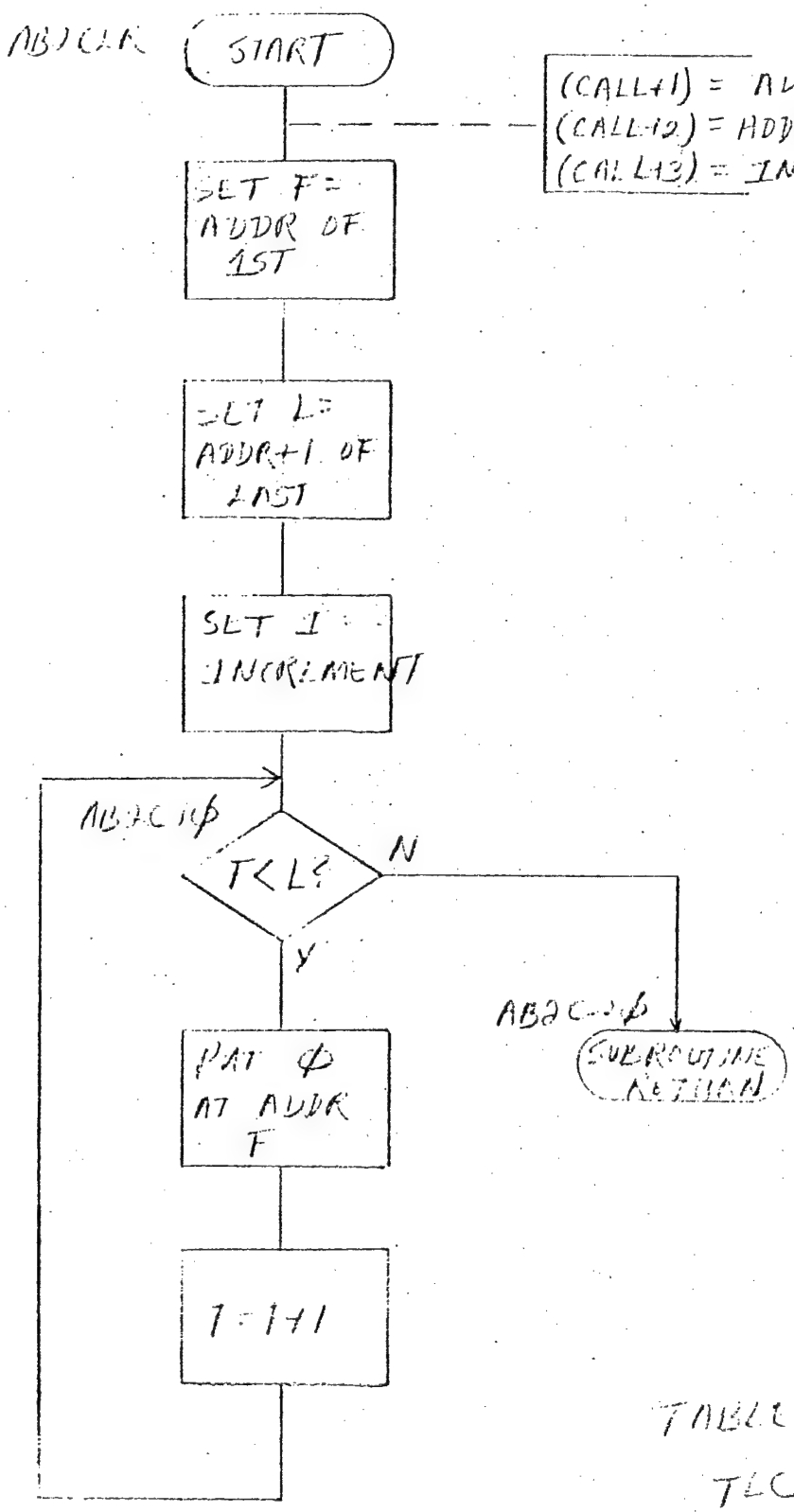


TABLE CLEARING ROUTINE

TLC 31 JAN 77

3.7 SUBPROGRAM LIMITATIONS

The ABI Management 1 and 2 Functional Groups shall have the following limitations.

1. Analysis resources shall be limited to:
   a) 8 AMT entries
   b) 8 ABI double buffers
   c) TBD 4-word data blocks
2. Buffer full processing (ABRDR) must be performed immediately after receipt of the buffer full interrupt to prevent the losing of PDW's.
3. Aborted analyses shall not be restarted. A null analysis return message shall be output. However, since only update analyses can be aborted, they will be restarted with the processing of the next Sorter update message.

3.8 INTERFACE DESCRIPTION

The interfaces of the ABI Management 1 and 2 functional groups are shown in Figure 5, and the following interface diagrams. A complete list of all subroutines and drivers in the ABI Management 1 and 2 Functional Groups is shown in Table 4.

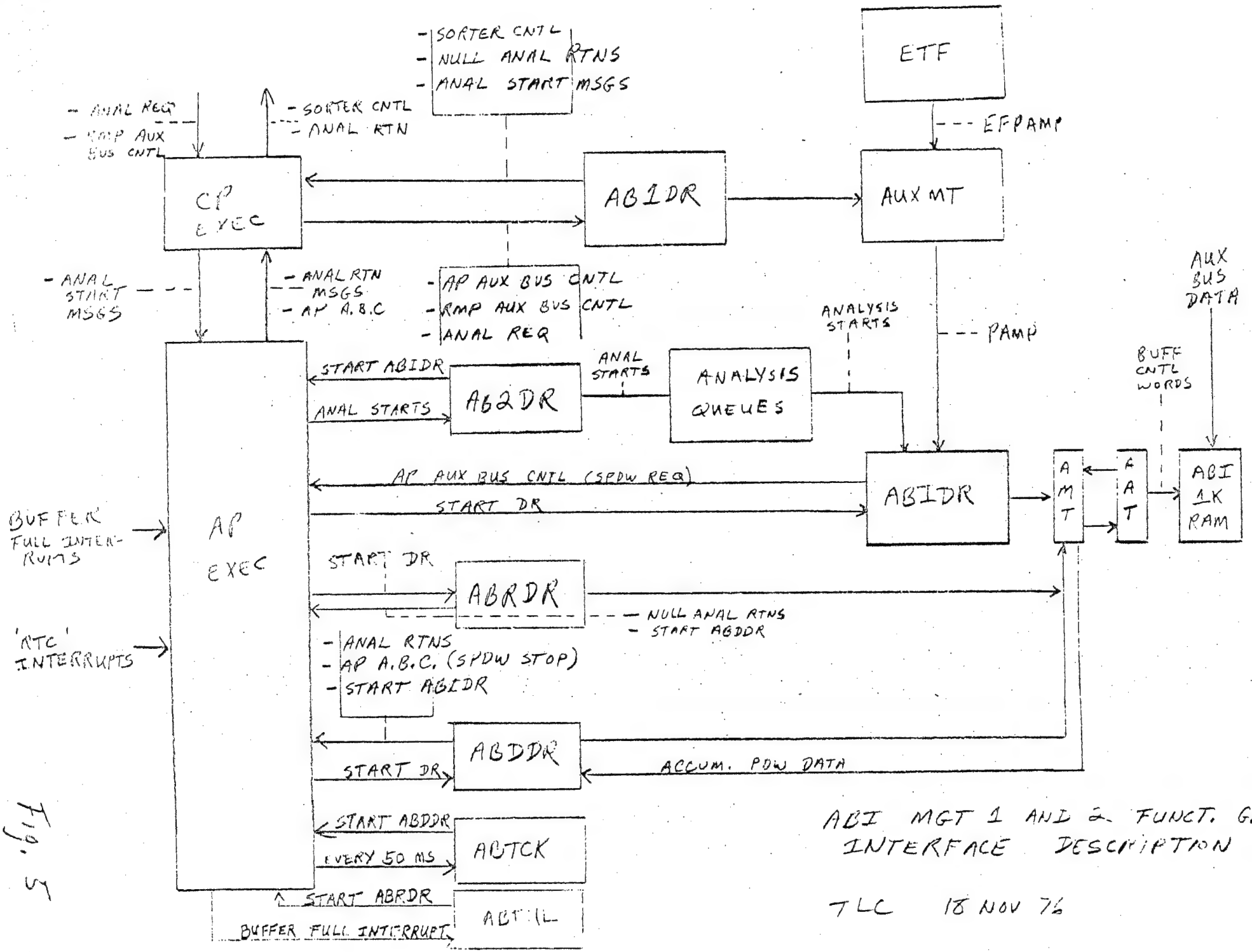
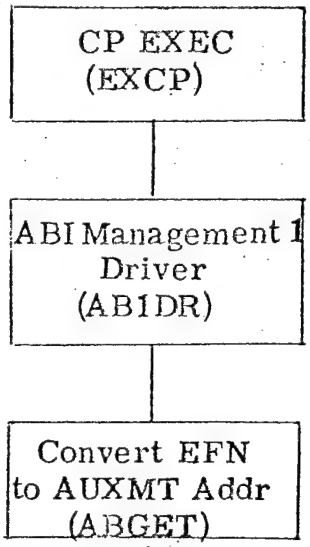


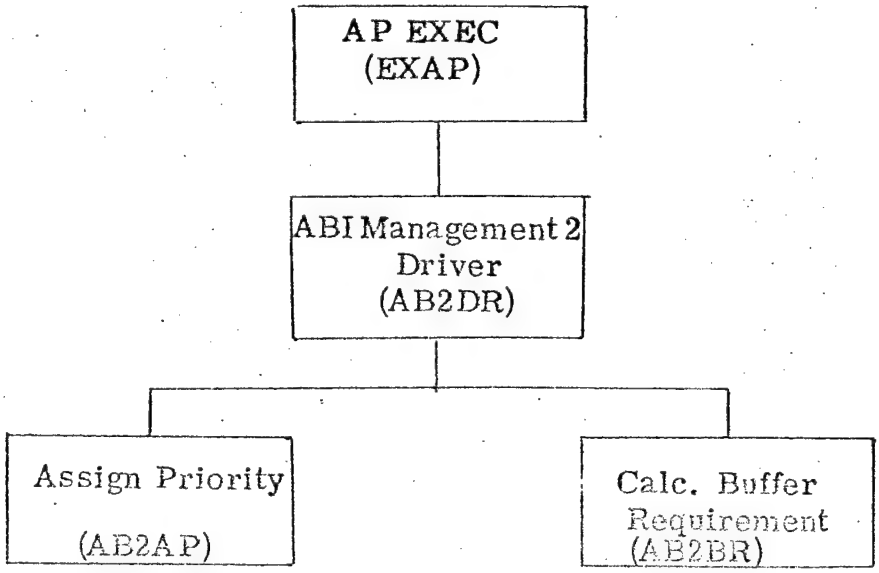
Fig. 5

ABI MGT 1 AND 2 FUNCT. GPS. INTERFACE DESCRIPTION

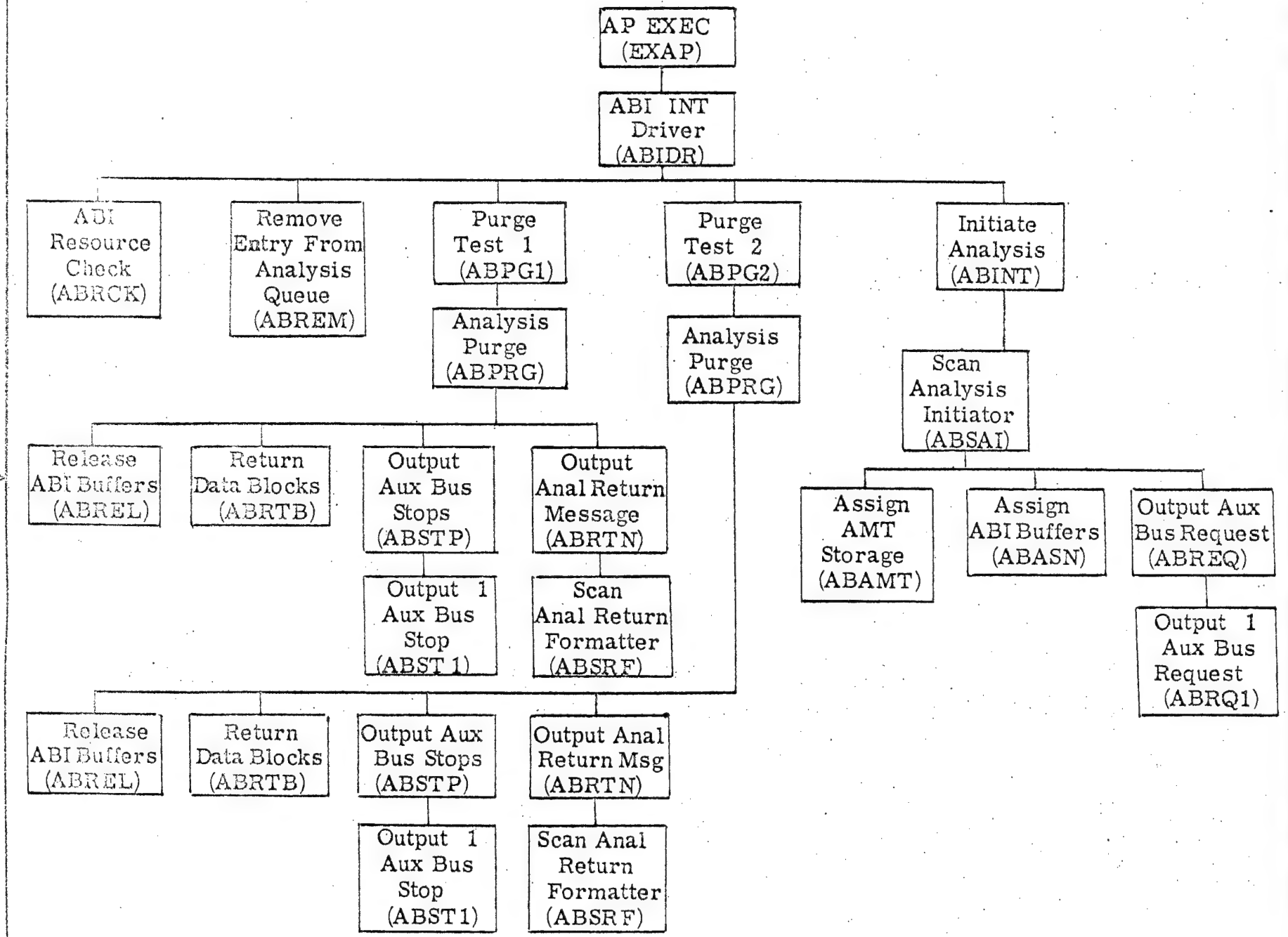
TLC 18 NOV 76



Interface Diagram - ABI Management 1 Driver



Interface Diagram - ABI Management 2 Driver



Interface Diagram - ABI Initialization Driver

RAYTHEON

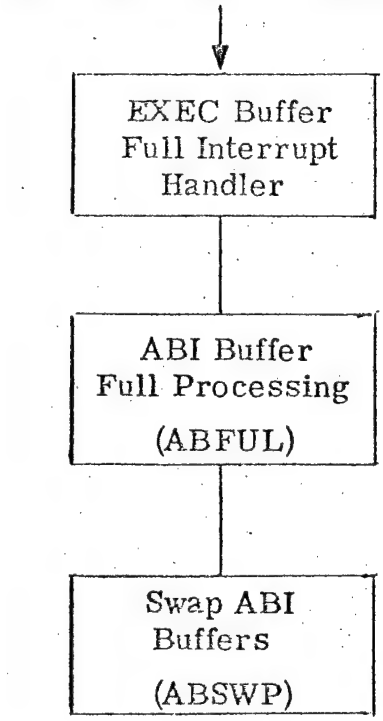
RAYTHEON COMPANY
LEXINGTON, MASS. 02173

CODE IDENT NO. 49956

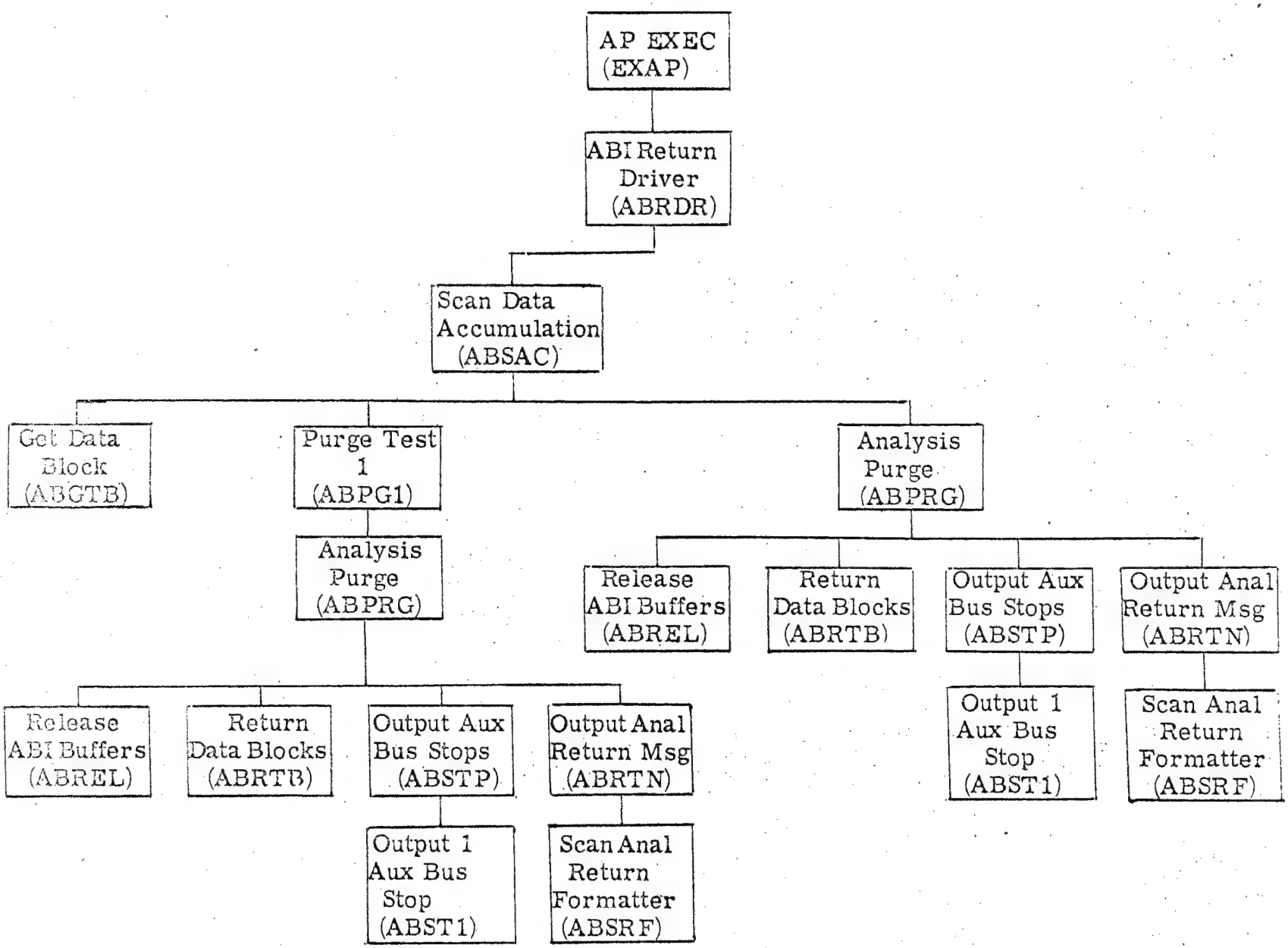
SPEC NO. 53959-GT-0754

SHEET 91 OF 97

REV



Interface Diagram - ABI Buffer Full Processing



Interface Diagram - ABI Return Driver

RAYTHEON

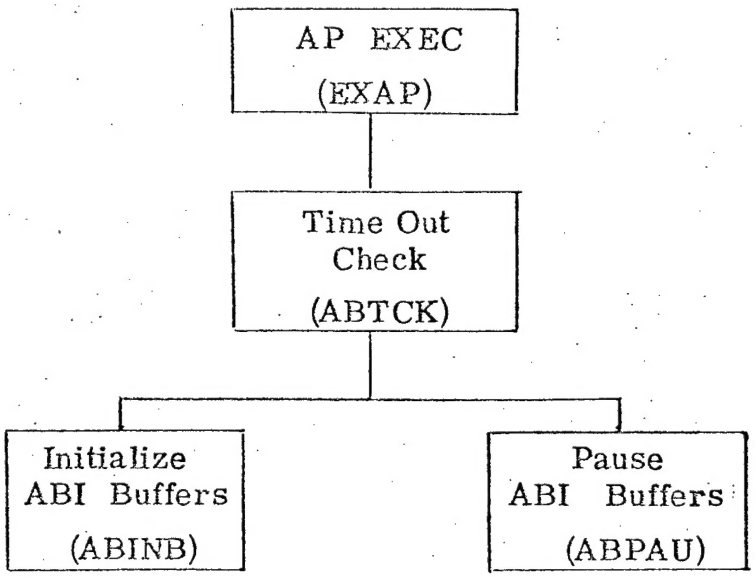
RAYTHEON COMPANY
LEXINGTON, MASS. 02173

CODE IDENT NO 49956

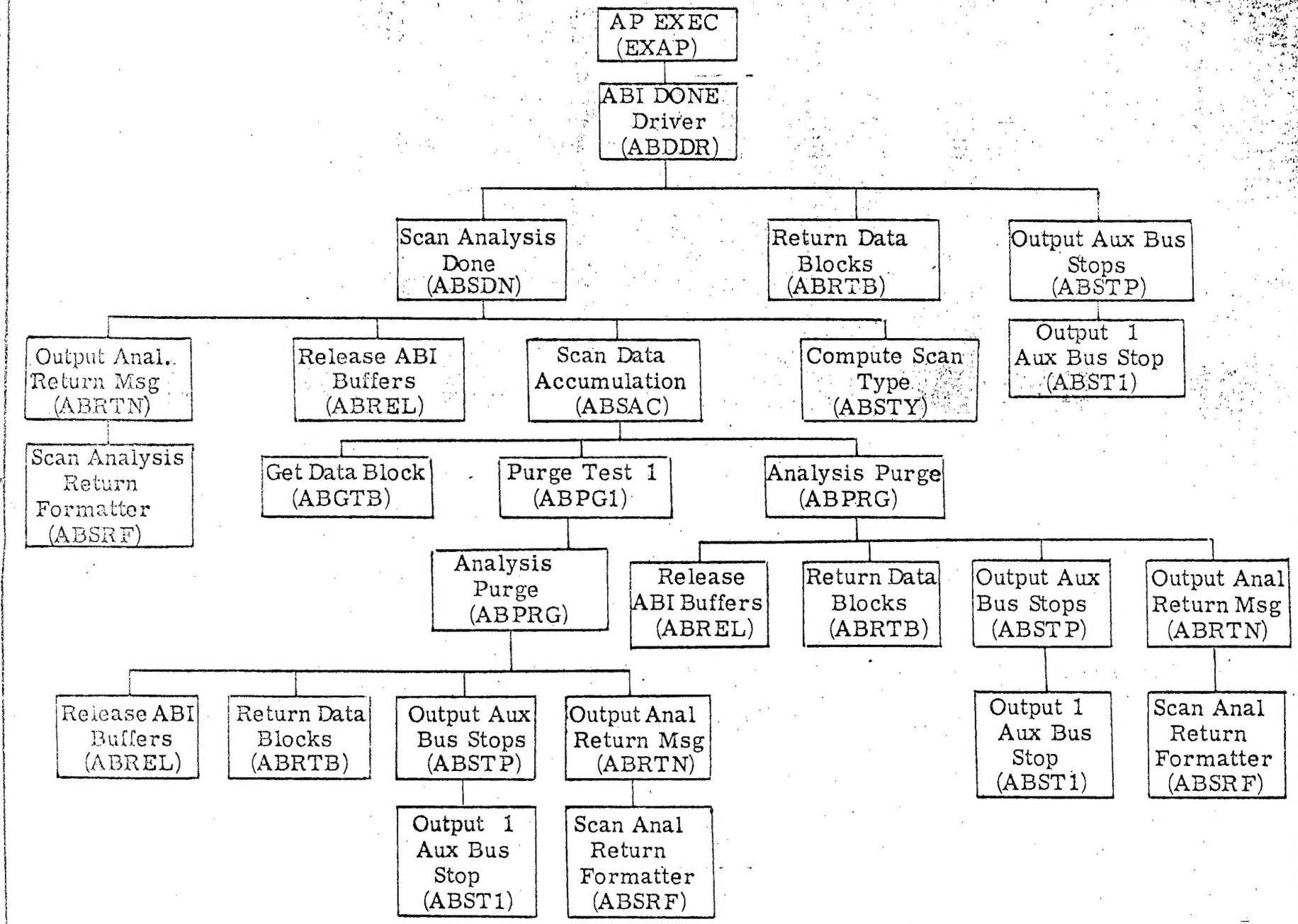
SPEC NO. 53959-CIT-0754

SHEET 93 OF 97

REV



Interface Diagram - Time Out Check



Interface Diagram – ABI Done Driver

RAYTHEON COMPANY
LEXINGTON, MASS. 02173

CODE IDENT NO. 49956

SPEC NO. 53959-GT-0754

SHEET 95 OF 97

REV

TABLE 4

ABI Management 1 and 2 Drivers and Subroutines

Mnemonic	Name
AB1DR	ABI Management 1 Driver
AB2DR	ABI Management 2 Driver
AB2AP	Assign Priority to Analysis Start Message
AB2BR	Calculate Buffer Requirement (for 1 Analysis)
AB2IN	ABI Initialization
ABAMT	Assign AMT Storage
ABASN	Assign ABI Buffers (to 1 Analysis)
ABDDR	ABI Done Driver
ABFUL	ABI Buffer Full Processing
ABGET	Convert EFN to AUXMT Address
ABGTB	Get Data Block (from free data blocks queue)
ABIDR	ABI Initialization Driver
ABINB	Initialize ABI Buffers (for 1 Analysis)
ABINT	Initiate Analysis
ABPAU	Pause ABI Buffers (for 1 Analysis)
ABPG1	Purge Test 1
ABPG2	Purge Test 2
ABPRG	Analysis Purge
ABRCK	ABI Resource Check
ABRDR	ABI Return Driver
ABREL	Release ABI Buffers (for 1 Analysis)
ABREM	Remove Entry from Analysis Queue
ABREQ	Output Aux Bus Request Messages
ABRQ1	Output 1 Aux Bus Request Message
ABRTB	Return Data Blocks (to free data blocks queue)
ABRTN	Output Analysis Return Message
ABSAC	Scan Data Accumulation

TABLE 4

(Continued)

Mnemonic	Name
ABSAI	Scan Analysis Initiator
ABSDN	Scan Analysis Done
ABSRF	Scan Analysis Return Formatter
ABST1	Output 1 Aux Bus Stop Message
ABSTP	Output Aux Bus Stop Messages
ABSTY	Scan Type Calculation
ABSWP	Swap ABI Buffers
ABTCK	Time Out Check